



Das Atari-Spielebuch für 600 XL/800 XL

James/Gee/Ewbank

BASIC

HC Mein Home-Computer



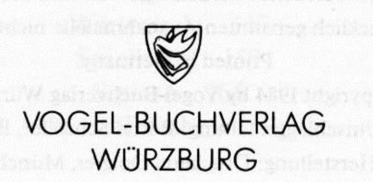
Mike James, S. M. Gee, Kay Ewbank

Das Atari-Spielebuch für 600 XL / 800 XL

HC – Mein Home-Computer

Mike James, S. M. Gee, Kay Ewbank

Das Atari-Spielebuch für 600 XL / 800 XL



Deutsche Übersetzung und Neubearbeitung:
Hans-Walter Beilstein, Bensheim

ISBN 3-8023-0788-7
1. Auflage. 1984
Dieses Buch ist eine Übersetzung der englischen Originalausgabe
«The Atari Book of Games».

Erstausgabe 1983 by Granada Publishing.

Printed in Germany.

Umschlaggestaltung: Bernd Schröder, Böhl
Herstellung: Druckerei Dotzler, München

Inhaltsverzeichnis

Einleitung

Die vorliegenden 21 Spiele sind ohne Änderungen auf jedem Atari lauffähig; Programm eintippen und los geht's! Viel Spaß!

Da die Programme «Fang den Quark», «Raketenabwehrschlacht», «Die Schatzinsel» und «Ataris Talkshow» sehr viel Speicherplatz benötigen, können bei Rechnern, die mit 16-K-RAM ausgestattet sind, eventuell vorhandene Diskettenlaufwerke wegen des Speicherbedarfs des DOS (Disk-Operating-System = Disketten-Betriebssystem) nicht benutzt werden. Wenn keine Speichererweiterung möglich ist, müssen die Diskettenlaufwerke abgeschaltet und die vier Programme auf Kassette gespeichert werden.

Die Spiele sind in BASIC geschrieben. Mit dieser Art der Programmierung und der Form der Originallistings sollen zwei Dinge erreicht werden. Dazu müssen die Programme selbst eingegeben werden, auch wenn es einige Mühe macht. Ein Anfänger wird so sehr schnell mit Syntax und Aufbau der Programmiersprache BASIC vertraut. Der erfahrene, fortgeschrittene BASIC-Programmierer wird nützliche Tips und Tricks kennenlernen, die auch in eigenen Programmen sehr gute Dienste leisten. Atari-BASIC ist nicht einfach, aber ungemein leistungsfähig und steht den anderen, auf dem Markt befindlichen BASIC-Versionen nicht nach. Die hier abgedruckten Programme sind eine eindrucksvolle Bestätigung dafür.

Die in diesem Buch gesammelten Spiele sind für die ganze Familie geeignet. Jedes Programm ist ausführlich beschrieben. Um die Qual der Wahl zu erleichtern, ist bei allen Spielen eine Bildschirmdarstellung abgedruckt. Natürlich kann das den eindrucksvollen Farbgrafiken des Atari nicht gerecht werden. Es gibt leider noch keine Möglichkeit, dem Leser die Geräuschkulisse der Spiele zu vermitteln.

ZU DIESEM BUCH UND SEINEN SPIELEN

Es ist schwer, jedes der 21 Spielprogramme genau einer Kategorie zuzuordnen oder in eine Rangfolge zu bringen. Etwa 2/3 der Spiele arbeiten mit bewegter Grafik. Bei Spielen wie «Invasion der Außerirdischen», «Squash» und «Schlittenfahrt» handelt es sich um Variationen bekannter und beliebter Spiele. Spiele wie «Die Schlucht», «Wächter der Herde» oder «Achtung – fertig – los!» sind neu und daher unbekannt. Sie sind aber so reizvoll und spannend, daß sie sicher bald zu den Favoriten gehören. «Raketenschlacht» und «Raketenabwehrschlacht» gehören zu den bekannten Feindabwehrspielen, heben sich aber durch die eingebauten Spezialeffekte deutlich ab. «Die Schatzinsel» ist mit eindrucksvoller Grafik ausgestattet und setzt ein gutes Gedächtnis voraus. «Fang den Quark» ist ein Brettspiel, bei dem man auf einem 8 x 8-Feld gegen den Atari spielt. Auch einige traditionelle Spiele sind dabei, die sich auf dem Computer hervorragend spielen lassen, z.B. «Atari-Würfelspiel», «Atari-Mühle» und «Wortpuzzle». Mit «Ataris Talkshow» wird ein ganz besonders interessantes Spiel vorgestellt: Es ist eine Unterhaltung zwischen Mensch und Computer in englischer Sprache.

ZU DEN PROGRAMMEN

Zum besseren Verständnis sind alle Programme strukturiert und die Unterprogramme (Subroutines) aufgelistet, damit deutlich wird, «wo, was und warum etwas» geschieht. Spezielle Tricks sind ausführlich erklärt, weitere Tips stehen in den Hinweisen zu jedem Programm. Wer solche Programmteile in eigenen Programmen verwendet, wird sehen, wie effektiv ein Atari programmiert werden kann.

Um den Leser zu eigenem Programmieren anzuregen, sind sämtliche Spiele in der Programmiersprache BASIC geschrieben. Zwar können solche Spiele nicht so schnell laufen wie die in Maschinensprache geschriebenen Spiele der Spielekassetten, aber Programmieren in Maschinensprache ist bedeutend schwieriger als in BASIC, besonders, wenn man noch nicht zu den Profis gehört.

Die Spielprogramme sind in Modultechnik aufgebaut. Sämtliche Berechnungen, Bildschirmdarstellungen usw. sind als eigenständige Unterprogramme (Moduln) vorhanden, die von einem relativ kurzen Hauptprogramm aufgerufen werden. So ist das von der Programmiersprache Pascal her bekannte Prozedurkonzept angewendet worden.

PROGRAMMLISTINGS

Bei den Listings in diesem Buch handelt es sich um die Originallistings, mit denen die Spiele getestet wurden. Es wurde bewußt auf eine Neunumerierung der Programmzeilen verzichtet (und «krumme» Zeilennummern in Kauf genommen), um Fehler bei dieser Aktion zu vermeiden. Die Programme sollten also genauso eingegeben werden, wie sie hier abgedruckt sind.

Zur besseren Lesbarkeit der Listings ist der Zeilenumbruch so gewählt, daß keine Wörter auseinandergerissen werden. Weiter wurden zwischen die einzelnen Programmteile Leerzeilen eingefügt.

Alle Zeichen, die im Grafik-Modus einzugeben sind, sind zwischen die spitzen Klammern < und > gesetzt. So bedeutet ein <R>, daß der Buchstabe R im Grafik-Modus einzugeben ist (siehe Rubrik Hinweise bei den Listings).

Um Verwechslungen zwischen der Ziffer Null (0) und dem Großbuchstaben «O» zu vermeiden, wurde weitgehend auf die Verwendung des großen «O» verzichtet. Bei der Eingabe ist darauf zu achten, daß diese Zeichen nicht verwechselt werden, weil der zum Ausdrukken der Listings verwendete Atari-Typenwalzendrucker die Null ohne Schrägstrich darstellt.

Um Schwierigkeiten beim Eingeben der Programme zu vermeiden, sind die Umlaute durch ae, oe usw. ersetzt worden. Sollen Um-

laute verwendet werden, ist zu beachten, daß im Grafikmodus u. U. statt der Umlaute Grafikzeichen ausgegeben werden.

Im Grafikmodus werden Groß- und Kleinbuchstaben in unterschiedlichen Farben ausgegeben. In den Listings sind daher einige Wörter nur in Groß- oder nur in Kleinbuchstaben geschrieben worden, um diesen Effekt zu erzielen. Ebenso ist bei Textzeilen, die über den rechten Bildschirmrand hinausgehen, teilweise auf einen neuen Druckbefehl verzichtet worden. Durch Einfügen oder Weglassen von Leerzeichen wurde eine entsprechende Bildschirmformatierung erreicht.

FEHLER . . .

Wie erwähnt, handelt es sich bei den abgedruckten Listings ausnahmslos um Originallistings lauffähiger Programme. Ein Programm, das ohne jede Änderung eingegeben wurde, sollte auf Anhieb funktionieren. Gerade beim Abschreiben von Texten können sich aber sehr leicht Fehler einschleichen. Sollte ein Programm nicht laufen, ist das eingegebene Programm Zeile für Zeile mit dem Originallisting zu vergleichen. Wenn das Spiel nach der ersten Korrektur immer noch nicht läuft, sollte man einen Augenblick Pause machen und dann noch einmal vergleichen.

Bestimmt sind einige Fehler überhaupt nicht aufgefallen. So ist besonders auf «Null-Strings» zu achten, das sind Zeichenketten ohne Zeichen (z.B. A$ = ””), oder auf Strings (Zeichenketten) mit Leerzeichen (A$ = ” ”). Die Anzahl der Leerzeichen (Blanks) ist wichtig. In REM-Zeilen (Kommentarzeilen) der Listings ist dort, wo das Abzählen zu umständlich gewesen wäre, die richtige Zahl der Blanks angegeben. Weitere Fehlerquellen liegen in der Verwechslung der Zeichen ”.” und ”,” ”:” und ”;” ”<” und ”>” sowie der richtigen Anzahl von Klammern (besonders bei Mehrfachklammern).

Auf keinen Fall sollten die Zeilennummern geändert werden! Es entstehen regelrechte Fallgruben im Programm, in die man unweigerlich stürzt, wenn zu einer geänderten Zeilennummer gesprungen werden soll. Welche der jetzt existierenden Zeilennummern war die

richtige? Bitte auch in diesem Zusammenhang auf die Hinweise achten.

Wenn das Spiel mit einer Fehlermeldung abbricht, sollte nicht nur die in der Fehlermeldung angegebene Zeile überprüft werden. Fehler können auch in weit entfernten Programmteilen liegen, z.B ein falsch dimensioniertes Feld (Array). Stehen Sprunganweisungen in der Zeile, sollte geprüft werden, ob die angegebenen Zieladressen (=Zeilennummern) tatsächlich existieren.

JOYSTICKS

Die hier veröffentlichten Spiele sind für die Grundausstattung eines Atari-Computers vorgesehen, auf die Verwendung von Joysticks (Steuerknüppel, Paddles) wurde daher in den Programmen verzichtet.

Für erfahrene Programmierer dürfte es jedoch keine Schwierigkeit bereiten, die Steuerung durch die Pfeiltasten auf Joysticks umzustellen. Geeignet sind die Programme «Schlittenfahrt», «Schwer auf Draht», «Invasion der Außerirdischen», «Tontaubenschießen» und «Squash». Bei den restlichen Spielen ist die Verwendung von Joysticks nicht möglich oder wenig sinnvoll.

SICHERHEITSKOPIEN

Es empfiehlt sich, Änderungen an einem Programm niemals am Original, sondern grundsätzlich an einer Kopie des Programms durchzuführen. Von allen lauffähigen Programmen sollte daher eine Sicherheitskopie (Back-up) angefertigt werden, die nicht angetastet wird und auf die man bei Schwierigkeiten zurückgreifen kann.

1 Schlittenfahrt

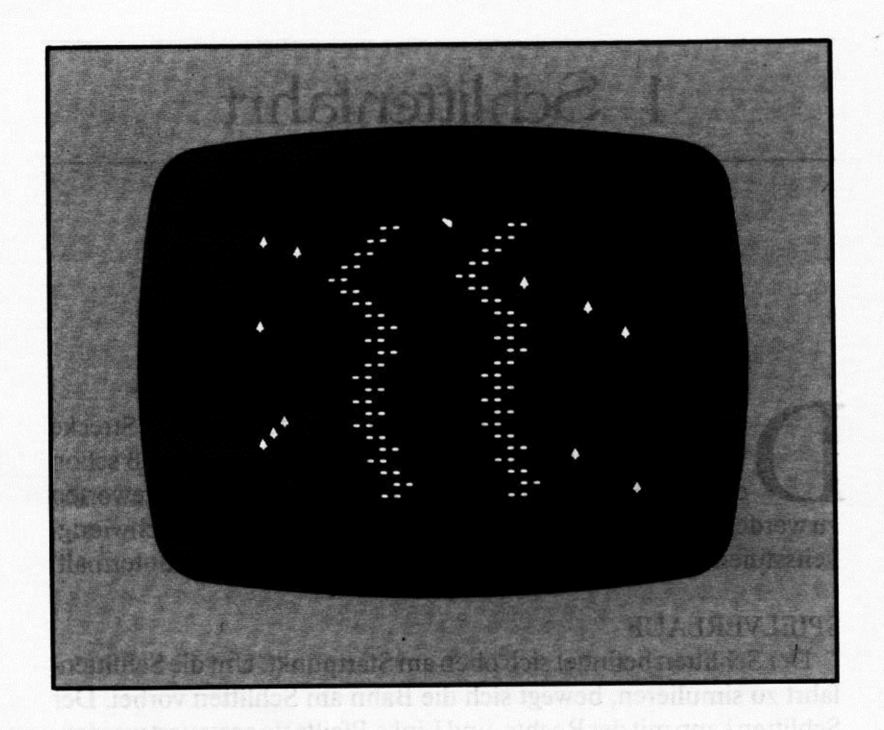
Das wilde Abenteuer einer rasanten Schlittenfahrt! Die Strecke windet sich durch einen Tannenwald, und man muß schon ein guter Slalomfahrer sein, um nicht aus der Bahn geworfen zu werden. Die Breite der Strecke kann gewählt werden, 5 Schwierigkeitsstufen stehen zur Verfügung; das Spiel wird akustisch untermalt.

SPIELVERLAUF

Der Schlitten befindet sich oben am Startpunkt. Um die Schlittenfahrt zu simulieren, bewegt sich die Bahn am Schlitten vorbei. Der Schlitten kann mit der Rechts- und Links-Pfeiltaste gesteuert werden. Die Bahnbegrenzung darf nicht berührt werden: Unfallgefahr! Vor dem Spielbeginn ist der Schwierigkeitsgrad zu wählen: Stufe 1 für «leicht» mit breiter Schlittenbahn, Stufe 5 ist am schwierigsten mit einer sehr schmalen Bahn.

HINWEISE

Die Zeilen in den Programmblöcken 1000 bis 1070, 1100 bis 1170, 1200 bis 1270 unterscheiden sich nur durch einzelne Zeichen. Hier sollten die guten Editiermöglichkeiten des Atari ausgenutzt werden, um Tippfehler zu vermeiden und Zeit zu sparen.



PROGRAMMAUFBAU

20	Umschalten auf Grafikmodus und Abschalten des Cursors
150	Beginn der Slalomstrecke
260	Hauptprogramm: Slalomfahrt
380	Ziel ist erreicht, das Ende der Strecke wird auf dem Bildschirm nach oben gerollt (Scrollingeffekt)
440	Meldung über Gewinn/Verlust
480	Ende des Spiels
560	Titelbild
670	Bewegung/Steuerung des Schlittens
770	Ausgabe der Tannenbäume auf dem Bildschirm
1000	Definition der Schlittendarstellung auf dem Bildschirm
1100	Definition der Bahnbegrenzung auf dem Bildschirm

1200 Definition der Tannenbaumdarstellung auf dem Bildschirm
1280 Umschaltung der Farben
2000 Tonerzeugung

ERLÄUTERUNGEN

Die Figurendarstellungen (Shapes) für Schlitten, Bahnbegrenzung und Tannenbaum werden als neue, selbst definierte Zeichen aufgerufen. Atari-BASIC hat hierfür zwar keine eigenen Befehle, die Definition eines neuen Zeichens ist jedoch nicht schwer. Die Form jedes Zeichens ist üblicherweise im ROM-Zeichensatz abgelegt. Da keine Reservezeichen zur Verfügung stehen, müssen einige der bereits vorhandenen Zeichen umdefiniert werden: $ = Schlitten, & = Bahnbegrenzung, % = Tannenbaum. Zunächst wird der gesamte Zeichensatz aus dem ROM- in den RAM-Bereich kopiert (Zeilen 50 bis 90). Durch Verändern entsprechender Speicherplätze kann nun für jedes Zeichen eine neue Form definiert werden (Zeilen 1000 bis 1270). Abschließend wird in den Zeilen 110 und 120 der ROM-Zeichensatz ab- und der RAM-Zeichensatz eingeschaltet.

Der Eindruck des fahrenden Schlittens entsteht, weil der Bildschirm am Schlitten, der sich nur nach rechts und links bewegen kann, vorbei emporgerollt wird. Das LOCATE-Kommando in Zeile 730 testet, ob der Schlitten gegen die Bahnbegrenzung fährt. Dazu wird geprüft, welches Zeichen sich am Bildschirm an der Stelle befindet, an der das Schlittensymbol als nächstes erscheinen soll. Wenn dieses Zeichen nicht den ASCII-Code 32 (Blank, Leerzeichen) hat, ist man (leider) aus der Bahn geflogen.

EIGENE ERGÄNZUNGEN

Es sollte versucht werden, das Programm durch eine «charakteristische» Melodie zu ergänzen. Im Spiel «Pferderennen» findet sich ein Beispiel hierzu.

```
10 REM Schlittenfahrt


20 GRAPHICS 0
30 GOSUB 560
40 DIM A$(1)
50 CH=(PEEK(106)-8)*256
60 CHORG=(PEEK(756)*256)
70 FOR I=0 TO 511
80 POKE CH+I,PEEK(CHORG+I)
90 NEXT I
100 GOSUB 1000
110 POKE 756,CH/256
120 POKE 752,1
.
150 YB=0
160 X=INT(RND(0)*10)+5
170 XB=INT(X+D/2)
175 PRINT CHR$(125)
180 FOR Y=1 TO 23
190 POSITION X,23:PRINT "&&";:POSITION X+D,23:PRINT "&&"
200 GOSUB 770
210 X=X+SGN(RND(0)-0.5)
220 IF X>29 THEN X=29
230 IF X<1 THEN X=1
240 NEXT Y
250 POSITION XB,YB:PRINT "$";

260 FOR Z=1 TO 500:NEXT Z
270 F=0
280 FOR Z=1 TO 300
290 X=X+SGN(RND(0)-0.5)
300 IF X>29 THEN X=29
310 IF X<1 THEN X=1
320 POSITION X,23:PRINT "&&";:POSITION X+D,23:PRINT "&&"
330 GOSUB 770
340 GOSUB 670
350 IF F=1 THEN GOTO 460
370 NEXT Z

380 FOR Z=1 TO 30
390 POSITION 19,23:PRINT
400 GOSUB 670
410 IF F=1 THEN GOTO 460
430 NEXT Z
```

```
440 POSITION 1,19:PRINT "BRAVO! TOLLE LEISTUNG! ZIEL ERREICHT"
450 GOTO 480
460 POSITION 1,19:PRINT "LEIDER AUS DER BAHN GEFLOGEN!"
470 GOSUB 2000
480 POSITION 1,20:PRINT "NOCH EIN VERSUCH (J/N)";
520 INPUT A$
530 IF A$="J" OR A$="j" THEN RUN
540 GRAPHICS 0
550 STOP


560 PRINT CHR$(125)
580 PRINT "           Schlittenfahrt":REM 11 Leerzeichen
589 POSITION 2,5:PRINT "Die Schlittenbahn ist sehr steil und"
590 PRINT "geht durch einen dichten Wald"
591 PRINT :PRINT "Der Schlitten ist mit den Tasten":PRINT
592 PRINT "  <== (Linkspfeil)    und"
593 PRINT "  ==> (Rechtspfeil)"
600 PRINT :PRINT "zu steuern."
620 POSITION 2,18:PRINT "Welcher Schwierigkeitsgrad -"
630 PRINT " <1> = leicht"
640 PRINT " <5> = schwer":PRINT
641 PRINT "Zahl von 1 bis 5 eingeben: ";:INPUT D
642 IF D<1 OR D>5 THEN GOTO 641
650 D=11-D
660 RETURN

670 REM Schlitten
680 A=PEEK(764)
685 POKE 764,255
710 IF A=6 THEN XB=XB-1
720 IF A=7 THEN XB=XB+1
730 LOCATE XB,YB,Q:IF Q<>32 THEN F=1
740 POSITION XB,YB:PRINT "$"
750 RETURN

770 REM Baum
780 IF RND(0)<0.4 THEN GOTO 860
790 K=SGN(RND(0)-0.5)
800 XT=X+K
810 XT=INT(X+D/2+(K*(RND(0)*5+D)))
820 IF XT<0 OR XT>38 THEN RETURN
840 POSITION XT,23:PRINT "%";
860 RETURN
```

```
1000 POKE CH+(ASC("$")-32)*8+0,32
1010 POKE CH+(ASC("$")-32)*8+1,40
1020 POKE CH+(ASC("$")-32)*8+2,232
1030 POKE CH+(ASC("$")-32)*8+3,252
1040 POKE CH+(ASC("$")-32)*8+4,126
1050 POKE CH+(ASC("$")-32)*8+5,62
1060 POKE CH+(ASC("$")-32)*8+6,30
1070 POKE CH+(ASC("$")-32)*8+7,14

1100 POKE CH+(ASC("&")-32)*8+0,16
1110 POKE CH+(ASC("&")-32)*8+1,16
1120 POKE CH+(ASC("&")-32)*8+2,16
1130 POKE CH+(ASC("&")-32)*8+3,16
1140 POKE CH+(ASC("&")-32)*8+4,16
1150 POKE CH+(ASC("&")-32)*8+5,16
1160 POKE CH+(ASC("&")-32)*8+6,124
1170 POKE CH+(ASC("&")-32)*8+7,124

1200 POKE CH+(ASC("%")-32)*8+0,24
1210 POKE CH+(ASC("%")-32)*8+1,24
1220 POKE CH+(ASC("%")-32)*8+2,60
1230 POKE CH+(ASC("%")-32)*8+3,60
1240 POKE CH+(ASC("%")-32)*8+4,126
1250 POKE CH+(ASC("%")-32)*8+5,126
1260 POKE CH+(ASC("%")-32)*8+6,24
1270 POKE CH+(ASC("%")-32)*8+7,24

1280 SETCOLOR 1,0,0
1290 SETCOLOR 2,0,14
1999 RETURN

2000 SOUND 0,100,10,10
2010 FOR Q=1 TO 50
2020 SOUND 0,0,0,0
2025 NEXT Q
2030 RETURN
```

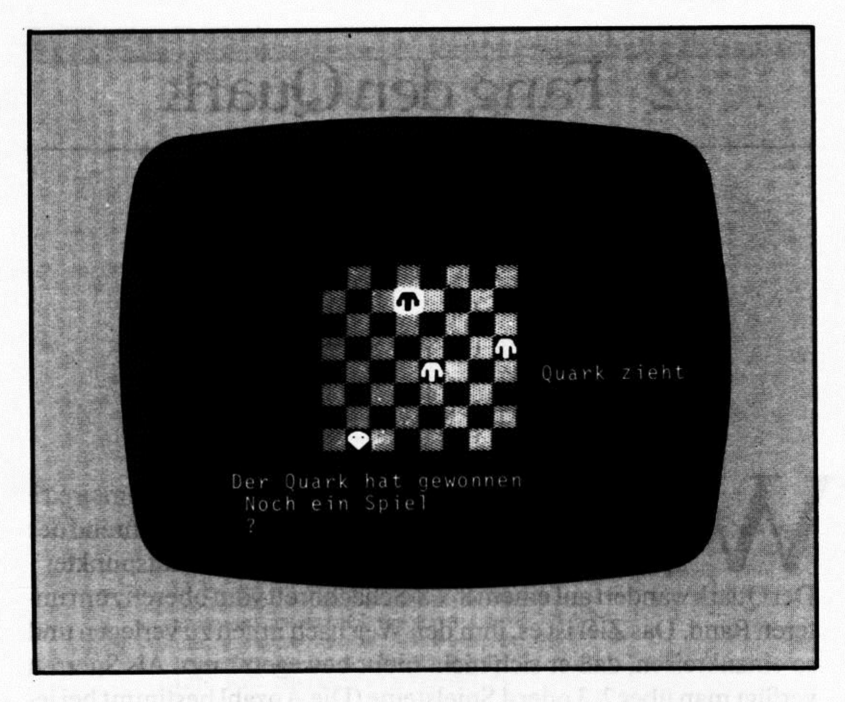
2 Fang den Quark

Was in aller Welt ist ein «Quark»? Nun, ausnahmsweise nichts zu essen, sondern ... - aber das soll man während des Spiels selbst herausfinden! Hier einige Anhaltspunkte: Der Quark wandert auf einem 8 x 8-Schachbrett vom oberen zum unteren Rand. Das Ziel ist es, ihm den Weg nach unten zu verlegen und so einzukreisen, daß er sich nicht mehr bewegen kann. Als Sperren verfügt man über 2, 3 oder 4 Spielsteine (Die Anzahl bestimmt bei jedem Spiel der Atari), die nur diagonal nach oben (!) um ein Feld je Spielzug bewegt werden können. Der Quark zieht zwar auch nur diagonal, aber vorwärts und rückwärts!

SPIELVERLAUF

Zu Beginn des Spiels liegen die Spielsteine in der untersten Reihe des Schachbretts, der Quark lauert am oberen Rand und wartet auf den ersten Zug. Mit dem invers dargestellten Spielstein kann gezogen werden. Durch Drücken der CONTROL-Taste zusammen mit einer beliebigen anderen Taste kann auf einen anderen Spielstein umgeschaltet werden. Am besten ausprobieren!

Die Spielsteine werden mit den Rechtspfeil- und Linkspfeiltasten bewegt. Linkspfeil: 1 Feld diagonal nach links oben, Rechtspfeil: 1 Feld diagonal nach rechts oben. Nach jedem Spielzug erfolgt der Gegenzug des Quarks.



Sollte man während des Spiels die Ausweglosigkeit seiner Situation erkennen, kann man durch ein «R» anstelle einer Pfeiltaste aufgeben. Um sicherzugehen, daß es sich nicht um einen Tippfehler handelt, bittet der Computer, die Aufgabe des Spiels durch ein «J» zu bestätigen. Bei «N» für NEIN geht das Spiel weiter.

HINWEISE

Bei dem IF-Befehl in Zeile 1010 sind keine Vergleichsoperatoren angegeben. Der Befehl ist trotzdem vollständig und wird vom Atari richtig ausgeführt. Die Elemente des angesprochenen Feldes sind hier entweder = 1 oder = 0, und für den Computer ist das gleichbedeutend mit den Booleschen Werten «wahr» und «falsch». Bei der Eingabe des Programms bitte besonders auf die Leerzeichen achten! Je 2 Leerzeichen müssen zwischen den Anführungsstrichen

in den Zeilen 2000 und 5500 stehen, je 2 Leerzeichen vor bzw. hinter den Dollarzeichen in den Zeilen 7020 und 7060.

PROGRAMMAUFBAU

20	Initialisierung der Variablen
50	Hauptteil des Programms
1000	Berechnung der Spielzüge des Quarks
5000	Bewegung der Steine, Prüfung, ob Zug erlaubt ist
6000	Auswahl des zu rückenden Spielsteins
6500	Inverse Darstellung eines Spielsteins
6600	Normale Darstellung eines Spielsteins
6700	Tonerzeugung
6800	Löschen der Spielzüge
7000	Zeichnen des Schachbretts
7500	Anfangspositionen der Spielsteine und des Quarks, Ausgabe auf dem Bildschirm
8000	Kopiert den Zeichensatz aus dem ROM- in den RAM-Bereich
8050	Definition eines Spielfeldes im RAM-Zeichensatz
8090	Quark-Darstellung im RAM-Zeichensatz
8420	Spielstein im RAM-Zeichensatz
8920	Umschalten des Atari auf den nun im RAM-Bereich definierten neuen Zeichensatz und Abschalten des Cursors
9000	Ende des Spiels

ERLÄUTERUNGEN

Das Programm benutzt hier nur die Schwarzweiß-Grafik, wobei die Darstellung des Schachbretts in weißen und schraffierten Feldern erfolgt, um die Übersichtlichkeit zu erhöhen (Zeilen 8050 bis 8080).

```
10 REM Fang den Quark

50 GOSUB 8000
55 SETCOLOR 1,0,15
56 SETCOLOR 2,0,0
60 GOSUB 7000
70 H=INT(RND(0)*3)+2:GOSUB 7500
130 GOSUB 6000
140 GOSUB 1000
150 GOTO 130

1000 POSITION 26,12:PRINT #6;"QUARK ZIEHT "
1010 IF D(QI+2,QJ+2) AND D(QI+2,QJ) AND
     D(QI,QJ+2) AND D(QI,QJ) THEN GOTO 9000
1020 M=1
1030 GOSUB 3000
1035 IF QI+N<1 OR QI+N>8 THEN GOTO 1100
1040 IF D(QI+N+1,QJ+M+1) THEN GOSUB 2500
1050 IF D(QI+N+2,QJ+M+2)=1 AND D(QI+N,QJ+M+2)=1
     AND QJ<7 AND QJ>1 THEN M=-M
1100 IF D(QI+N+1,QJ+M+1) THEN GOSUB 2500
1110 IF OI=QI AND OJ=QJ THEN M=-M:
     N=SGN(RND(1)-0.5):GOTO 1035
1120 OI=QI:OJ=QJ
2000 POSITION QX,QY:PRINT #6;"  ";
2001 POSITION QX,QY+1:PRINT #6;"  ";
2010 QX=QX+N*2
2020 QY=QY+M*2
2030 POSITION QX,QY:PRINT #6;"23";
2031 POSITION QX,QY+1:PRINT #6;"45";
2040 D(QI+N+1,QJ+M+1)=2
2050 D(QI+1,QJ+1)=0
2060 QI=QI+N
2070 QJ=QJ+M
2080 IF QJ=8 THEN GOTO 9500
2090 RETURN
2500 N=-N
2510 IF D(QI+N+1,QJ+M+1)<>1 THEN RETURN
2520 M=-M
2525 IF QJ<4 THEN N=1
2530 IF D(QI+N+1,QJ+M+1)<>1 THEN RETURN
2540 N=-N
2550 RETURN
```

```
3000 N=(QI<5)-(QI>=5)
3005 R=RND(1)
3010 IF QJ>6 THEN RETURN
3020 IF R>0.5 AND QI>3 THEN GOTO 3050
3025 IF QI>7 THEN GOTO 3035
3030 IF (D(QI+3,QJ+3)=0 OR D(QI+2,QJ+3)=0)
     AND D(QI+2,QJ+2)=0 THEN N=-1:RETURN
3035 IF QI<4 OR R>0.5 THEN RETURN
3050 IF (D(QI-3,QJ+3)=0 OR D(QI-2,QJ+3)=0)
     AND D(QI-2,QJ+2)=0 THEN N=1:RETURN
3060 IF R>0.5 THEN GOTO 3025
3070 RETURN
5000 A(HM)=A(HM)+M
5010 B(HM)=B(HM)-1
5020 IF D(A(HM)+1,B(HM)+1)<>0 THEN GOTO 5100
5050 D(A(HM)-M+1,B(HM)+2)=0
5060 D(A(HM)+1,B(HM)+1)=1
5070 GOTO 5500
5100 A(HM)=A(HM)-M
5110 B(HM)=B(HM)+1
5120 GOSUB 6700
5130 GOTO 6000
5500 POSITION X(HM),Y(HM):PRINT #6;"  ";
5501 POSITION X(HM),Y(HM)+1:PRINT #6;"  ";
5510 Y(HM)=Y(HM)-2
5520 X(HM)=X(HM)+M*2
5530 POSITION X(HM),Y(HM):PRINT #6;"68";
5531 POSITION X(HM),Y(HM)+1:PRINT #6;"79";
5540 GOSUB 6500
5550 RETURN

6000 GOSUB 6700
6007 POSITION 26,12:PRINT #6;"BITTE ZIEHEN"
6010 GET #2,M
6025 IF M=82 THEN GOTO 6100
6030 IF M=43 THEN M=-1:GOTO 5000
6040 IF M=42 THEN M=1:GOTO 5000
6050 GOSUB 6200
6060 GOTO 6000
6100 POSITION 26,14:PRINT #6;"ABBRECHEN (J)"
6105 GET #2,A
6110 IF A=74 THEN GOTO 9500
6120 POSITION 26,14:PRINT #6;"            "
6121 REM 12Leerzeichen
6130 GOTO 6000
6200 GOSUB 6600
6210 HM=HM+1
6220 IF HM>H THEN HM=1
6230 GOSUB 6500
6240 RETURN
```

```
6500 POSITION X(HM),Y(HM)
6501 PRINT #6;CHR$(54+128);CHR$(56+128);
6510 POSITION X(HM),Y(HM)+1
6511 PRINT #6;CHR$(55+128);CHR$(57+128)
6520 RETURN

6600 POSITION X(HM),Y(HM):PRINT #6;"68";
6601 POSITION X(HM),Y(HM)+1:PRINT #6;"79";
6610 RETURN

6700 SOUND 0,80,10,10
6710 FOR I=1 TO 50
6720 NEXT I
6730 SOUND 0,0,0,0
6740 RETURN

6800 FOR I=1 TO 10
6810 FOR J=1 TO 10
6820 D(I,J)=0
6830 NEXT J
6840 NEXT I
6850 RETURN

7000 POSITION 8,3
7005 FOR I=1 TO 4
7010 FOR J=1 TO 8
7020 PRINT "  $$";
7030 IF J/4=INT(J/4) THEN PRINT :PRINT "      ";
7031 REM 6 Leerzeichen
7040 NEXT J
7050 FOR J=1 TO 8
7060 PRINT "$$  ";
7070 IF J/4=INT(J/4) THEN PRINT :PRINT "      ";
7071 REM 6 Leerzeichen
7080 NEXT J
7090 NEXT I
7100 RETURN

7500 FOR I=1 TO H
7510 X=I*4+6
7520 POSITION X,17:PRINT #6;"68";
7521 POSITION X,18:PRINT #6;"79";
7530 D(I*2+1,9)=1
7540 X(I)=X
7550 Y(I)=17
7560 HM=1
```

```
7570 A(I)=I*2
7580 B(I)=8
7590 NEXT I
7600 GOSUB 6500
7610 QI=5
7620 QJ=1
7630 QX=QI*2+6
7640 QY=3
7650 POSITION QX,QY:PRINT #6;"23"
7651 POSITION QX,QY+1:PRINT #6;"45";
7660 FOR I=1 TO 10
7670 D(I,1)=1
7680 D(1,I)=1
7690 D(10,I)=1
7700 D(I,10)=1
7710 NEXT I
7720 D(QI+1,QJ+1)=2
7730 OI=0
7740 OJ=0
7750 RETURN

8000 GRAPHICS 0
8001 POSITION 1,1:PRINT "EINEN AUGENBLICK BITTE"
8005 CH=(PEEK(106)-8)*256
8010 CHORG=(PEEK(756)*256)
8020 FOR I=0 TO 511
8030 POKE CH+I,PEEK(CHORG+I)
8040 NEXT I

8050 FOR Q=0 TO 7 STEP 2
8060 POKE CH+(ASC("$")-32)*8+Q,51
8070 POKE CH+(ASC("$")-32)*8+Q+1,204
8080 NEXT Q

8090 POKE CH+(ASC("2")-32)*8+0,0
8100 POKE CH+(ASC("2")-32)*8+1,0
8120 POKE CH+(ASC("2")-32)*8+2,7
8130 POKE CH+(ASC("2")-32)*8+3,31
8140 POKE CH+(ASC("2")-32)*8+4,63
8150 POKE CH+(ASC("2")-32)*8+5,115
8160 POKE CH+(ASC("2")-32)*8+6,115
8170 POKE CH+(ASC("2")-32)*8+7,127
8180 POKE CH+(ASC("3")-32)*8+0,0
8190 POKE CH+(ASC("3")-32)*8+1,0
8200 POKE CH+(ASC("3")-32)*8+2,224
8210 POKE CH+(ASC("3")-32)*8+3,248
```

```
8220 POKE CH+(ASC("3")-32)*8+4,252
8230 POKE CH+(ASC("3")-32)*8+5,206
8240 POKE CH+(ASC("3")-32)*8+6,206
8250 POKE CH+(ASC("3")-32)*8+7,254
8260 POKE CH+(ASC("4")-32)*8+0,127
8270 POKE CH+(ASC("4")-32)*8+1,63
8280 POKE CH+(ASC("4")-32)*8+2,31
8290 POKE CH+(ASC("4")-32)*8+3,15
8300 POKE CH+(ASC("4")-32)*8+4,7
8310 POKE CH+(ASC("4")-32)*8+5,3
8320 POKE CH+(ASC("4")-32)*8+6,1
8330 POKE CH+(ASC("4")-32)*8+7,0
8340 POKE CH+(ASC("5")-32)*8+0,254
8350 POKE CH+(ASC("5")-32)*8+1,252
8360 POKE CH+(ASC("5")-32)*8+2,248
8370 POKE CH+(ASC("5")-32)*8+3,240
8380 POKE CH+(ASC("5")-32)*8+4,224
8390 POKE CH+(ASC("5")-32)*8+5,192
8400 POKE CH+(ASC("5")-32)*8+6,128
8410 POKE CH+(ASC("5")-32)*8+7,0
8420 POKE CH+(ASC("6")-32)*8+0,0
8430 POKE CH+(ASC("6")-32)*8+1,0
8440 POKE CH+(ASC("6")-32)*8+2,15
8450 POKE CH+(ASC("6")-32)*8+3,31
8460 POKE CH+(ASC("6")-32)*8+4,31
8470 POKE CH+(ASC("6")-32)*8+5,63
8480 POKE CH+(ASC("6")-32)*8+6,55
8490 POKE CH+(ASC("6")-32)*8+7,115
8500 POKE CH+(ASC("7")-32)*8+0,115
8510 POKE CH+(ASC("7")-32)*8+1,115
8520 POKE CH+(ASC("7")-32)*8+2,115
8530 POKE CH+(ASC("7")-32)*8+3,115
8540 POKE CH+(ASC("7")-32)*8+4,3
8550 POKE CH+(ASC("7")-32)*8+5,3
8560 POKE CH+(ASC("7")-32)*8+6,3
8570 POKE CH+(ASC("7")-32)*8+7,0
8580 POKE CH+(ASC("8")-32)*8+0,0
8590 POKE CH+(ASC("8")-32)*8+1,0
8600 POKE CH+(ASC("8")-32)*8+2,240
8610 POKE CH+(ASC("8")-32)*8+3,248
8620 POKE CH+(ASC("8")-32)*8+4,248
8630 POKE CH+(ASC("8")-32)*8+5,252
8650 POKE CH+(ASC("8")-32)*8+6,220
8660 POKE CH+(ASC("8")-32)*8+7,206
8670 POKE CH+(ASC("9")-32)*8+0,206
8680 POKE CH+(ASC("9")-32)*8+1,206
8690 POKE CH+(ASC("9")-32)*8+2,206
8700 POKE CH+(ASC("9")-32)*8+3,206
8710 POKE CH+(ASC("9")-32)*8+4,192
```

```
8720 POKE CH+(ASC("9")-32)*8+5,192
8730 POKE CH+(ASC("9")-32)*8+6,192
8740 POKE CH+(ASC("9")-32)*8+7,0


8920 PRINT CHR$(225)
8930 POKE 756,CH/256
8940 POKE 752,1
8945 POSITION 1,1:PRINT "                     ":REM 21 Leerz.
8950 RETURN


9000 POSITION 1,20:PRINT "PRIMA, GEWONNEN!"
9010 GOTO 9900
9500 POSITION 1,20:PRINT "DER QUARK HAT GEWONNEN!"
9900 PRINT "NOCH EIN SPIEL (J/N) ";
9910 INPUT A$
9930 IF A$="J" THEN RUN
9940 GRAPHICS 0
9950 PRINT "AUF WIEDERSEHEN !!!"
```

3 Pferderennen

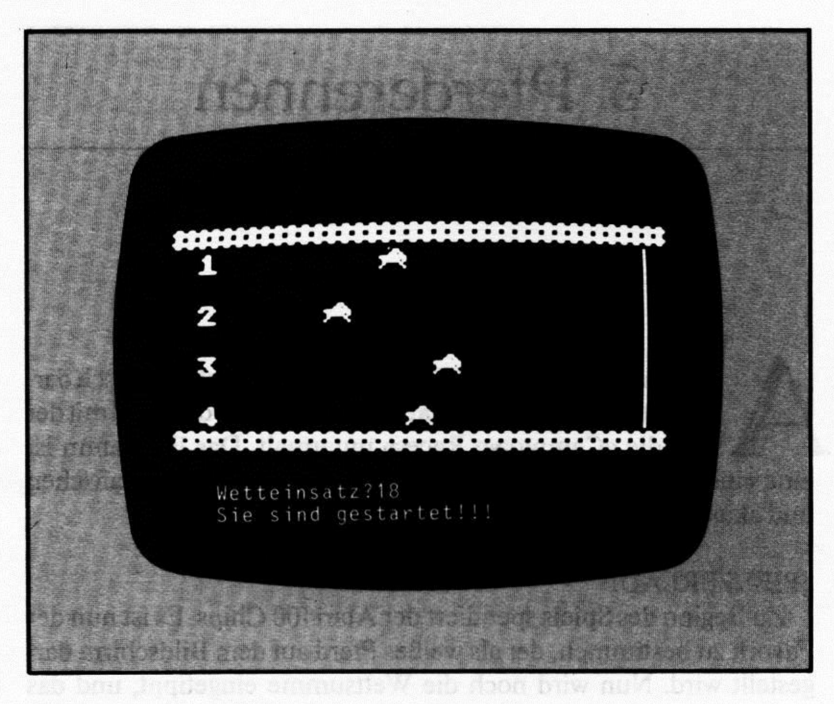
Auf dem Bildschirm findet ein rasantes, äußerst realistisch dargestelltes Pferderennen statt. Akustisch wird das Spiel mit der Melodie «Camptown Races» untermalt. Das Programm ist eine eindrucksvolle Demonstration der hervorragenden grafischen und akustischen Möglichkeiten des Atari-Computers.

SPIELVERLAUF

Zu Beginn des Spiels spendiert der Atari 100 Chips. Es ist nun der Favorit zu bestimmen, der als weißes Pferd auf dem Bildschirm dargestellt wird. Nun wird noch die Wettsumme eingetippt, und das wilde Rennen beginnt. Die Quote ist 5:1, d.h. es werden für 20 gesetzte Chips 100 ausbezahlt, falls der Favorit gewinnt. Die Finanzen werden vom Computer überwacht und Meldungen über Gewinn bzw. Verlust ausgegeben. Während das Rennen läuft, gibt es keine Möglichkeit, die Pferde zu beeinflussen.

HINWEISE

Beim Eingeben des Programms ist besonders auf die Punkte und Kommas der DATA-Statements ab Zeile 6000 zu achten. Fehler bedeuten hier nicht unbedingt den Abbruch des Programms, aber da es sich bei diesen Zeilen um die Noten der Melodie handelt, kann es bei Tippfehlern zu unschönen Dissonanzen kommen. Bei dem Zeichen zwischen den Anführungsstrichen in Zeile 220 handelt es sich um das Doppelkreuz «#» (Shift-3).



PROGRAMMAUFBAU

200 Darstellung des Pferdes
600 Unterprogramm-Aufruf für Melodie und Titelbild
610 Initialisierung der Variablen
1070 Rennstrecke
1560 Start des Rennens
2000 Titelbild
3000 Wetteinsatz
4000 Bewegung des Pferdes
5000 Spielt die gesamte Melodie
6000 Spielt genau eine Note der Melodie
7000 Bewegung der Beine des Pferdes
8000 Berechnung der Finanzen (Gewinn bzw. Verlust)
9000 Neues Rennen bzw. Ende des Spiels

ERLÄUTERUNGEN

Pferderennen ist das einzige Programm dieses Buchs, das eine komplette Melodie spielt. Die Noten sind in den DATA-Zeilen 6000 bis 6080 gespeichert. Es handelt sich um Zahlenpaare, wobei der erste Wert die Tonhöhe, der zweite die Tonlänge bestimmt. Typische Werte für Tonlängen sind 1 (Halbe-Note), 0.5 (Achtel-Note) oder 0.25 (Sechzehntel-Note). Der Wert -99 als Tonlänge wird in eine Pause umgesetzt. Das Zahlenpaar 99,99 in Zeile 6080 kennzeichnet das Ende der Melodie und veranlaßt den Computer, den DATA-Zeiger wieder auf das erste DATA-Element zu setzen, damit die Melodie im nächsten Durchgang wieder von vorn gespielt werden kann.

Zu Spielbeginn wird durch Aufruf des Unterprogramms 5000 die gesamte Melodie gespielt. Mit einem Trick ist die Melodie auch während des Rennens zu hören. Ein Computer kann nie mehrere Dinge gleichzeitig tun. Um während des Galopps die Melodie zu spielen, wird durch Aufruf des Unterprogramms 6000 zunächst nur ein Ton gespielt. Dabei rückt der DATA-Zeiger nach dem Lesen eines Wertes automatisch auf das nächste Element vor. Da hier kein RESTORE-Befehl verwendet wird (Rücksetzen des DATA-Zeigers auf das erste Element), wird bei dem nächsten READ sofort das richtige Wertepaar gelesen und als Ton ausgegeben. Nachdem eine Note gespielt worden ist, werden die Pferde ein Stück weiterbewegt, dann die nächste Note gespielt usw. Aufgrund der eingeschobenen Pferdebewegungen sind die Pausen zwischen den Tönen etwas größer. Die Bewegungen werden aber so schnell durchgeführt, daß der Eindruck einer Melodie erhalten bleibt. Wenn dieser Trick auch in anderen Programmen verwendet werden soll, ist darauf zu achten, daß solche Einschübe nicht zu lange dauern (Rechenzeit prüfen!), sonst sind zwar noch die Töne, aber keine Melodie mehr zu hören.

EIGENE ERGÄNZUNGEN

Wurde das Lied oft genug gehört, sollte eine andere Melodie als die hier vorgesehene eingegeben werden. Interessant wäre auch eine Änderung des Programms, die es erlaubt, seinen Favoriten zu lenken.

```
10 REM Pferderennen

200 GRAPHICS 2+16
210 CH=(PEEK(106)-8)*256
220 CHORG=(PEEK(756)*256)
230 FOR I=0 TO 511
240 POKE CH+I,PEEK(CHORG+I)
250 NEXT I
400 FOR I=0 TO 7
410 POKE CH+(ASC("&")-32)*8+I,128
420 NEXT I
500 POKE CH+(ASC("$")-32)*8+0,0
510 POKE CH+(ASC("$")-32)*8+1,12
520 POKE CH+(ASC("$")-32)*8+2,26
530 POKE CH+(ASC("$")-32)*8+3,255
540 POKE CH+(ASC("$")-32)*8+4,125
550 POKE CH+(ASC("$")-32)*8+5,66
560 POKE CH+(ASC("$")-32)*8+6,129
570 POKE CH+(ASC("$")-32)*8+7,0
580 POKE 756,CH/256

600 GOSUB 5000

610 TOTAL=100
620 FLAG=0
1000 DIM X(5),Y(5),A$(5)
1005 GRAPHICS 2
1010 SETCOLOR 0,0,0
1020 SETCOLOR 1,0,14
1030 SETCOLOR 2,12,4
1040 SETCOLOR 3,3,5
1050 SETCOLOR 4,12,4
1060 POKE 756,CH/256

1070 RESTORE
1200 FOR X=0 TO 18
1220 POSITION X,0:PRINT #6;"#"
1225 POSITION X,8:PRINT #6;"#"
1230 NEXT X
1300 FOR Y=1 TO 7
1310 POSITION 18,Y:PRINT #6;"&"
1320 NEXT Y
1500 FOR Y=1 TO 4
1520 X(Y)=2
1530 Y(Y)=Y*2-1
```

```
1540 POSITION X(Y)-1,Y(Y):PRINT #6;Y;CHR$(132)
1550 NEXT Y
1560 GOSUB 3000
1570 TEMPO=50
1710 GOSUB 4000
1720 GOTO 1710


2000 SETCOLOR 0,0,0
2005 SETCOLOR 3,3,5
2010 SETCOLOR 4,0,14
2020 POSITION 3,2
2050 PRINT #6;" P F E R D E"
2060 POSITION 5,4
2070 PRINT #6;" R E N N E N"
2080 RETURN


3000 PRINT "AUF WELCHES PFERD SOLL GESETZT WERDEN ";:INPUT B
3010 IF B<1 OR B>4 THEN PRINT "EIN SOLCHES PFERD GIBT ES NICHT!":GOTO 3000
3020 PRINT "GUTHABEN: ";TOTAL;" DM"
3030 PRINT "WETTEINSATZ: ";:INPUT M
3040 IF TOTAL-M<0 THEN PRINT "DAS GELD REICHT NICHT!":GOTO 3020
3050 TOTAL=TOTAL-M
3060 PRINT :PRINT "SIE SIND GESTARTET!!!"


4000 REM Pferdebewegungen
4010 Z=INT(RND(0)*4)+1
4050 POSITION X(Z),Y(Z):PRINT #6;" "
4110 X(Z)=X(Z)+(1+RND(0)*0.4)
4120 COL=132
4130 IF Z=B THEN COL=4
4140 POSITION X(Z),Y(Z):PRINT #6;CHR$(COL)
4145 GOSUB 7000
4160 IF X(Z)>16.5 THEN GOTO 8000
4170 GOSUB 6000
4180 IF T=99 THEN RESTORE
4190 RETURN

5000 GOSUB 2000
5005 TEMPO=50
5010 I=1
5015 GOSUB 6000
5020 IF T=99 THEN POSITION I+1,9:PRINT ;" ":RESTORE :RETURN
5030 POSITION I,9:PRINT #6;" ";CHR$(132)
5032 GOSUB 7000
5035 FOR Q=0 TO 10:NEXT Q
```

```
5040 I=I+0.3
5060 GOTO 5015


6000 DATA 72,.5,72,.5,72,.5,85,.5,72,.5,64,.5
6005 DATA 72,.5,85,.5,-99,.5,85,.5,96,1.5,85,.5,96,1
6010 DATA 72,.5,72,.5,85,.5,72,.5,64,.5,72,.5,85,.5
6020 DATA -99,.5,85,.25,96,.25,108,.25,96,.25
6025 DATA 85,.5,96,.5,108,1.5
6030 DATA -99,1,108,.75,108,.25,85,.5,72,.5,53,1.5
6040 DATA -99,.5,64,.75,64,.25,53,.5,64,.5,72,1.5
6050 DATA 85,.25,81,.25,72,.5,72,.5,85,.25,85,.25
6060 DATA 72,.25,72,.25,64,.5,72,.5,85,1
6070 DATA 96,.5,85,.5,81,.25,85,.5,96,.25,96,.25,108,1.5
6080 DATA 99,99
6090 READ P,T
6100 IF T=99 THEN RETURN
6110 T=T*TEMPO
6120 IF P=-99 THEN FOR Q=1 TO T:NEXT Q:RETURN
6130 SOUND 0,P,10,8
6140 FOR Q=1 TO T:NEXT Q
6150 SOUND 0,P,10,0
6160 RETURN


7000 IF FLAG=1 THEN GOTO 7500
7010 FLAG=1
7020 POKE CH+(ASC("$")-32)*8+6,40
7030 POKE 756,CH/256
7040 RETURN
7500 FLAG=0
7510 POKE CH+(ASC("$")-32)*8+6,129
7520 POKE 756,CH/256
7530 RETURN

8000 IF Z=B THEN PRINT "PRIMA, GEWINN: ";:TOTAL=TOTAL+INT(5*M)
8001 PRINT INT(5*M);" DM"
8010 IF Z<>B THEN PRINT "VERLUST ";M;" DM"
8020 IF TOTAL<=0 THEN PRINT "PECH GEHABT, PLEITE!":GOTO 9020
```

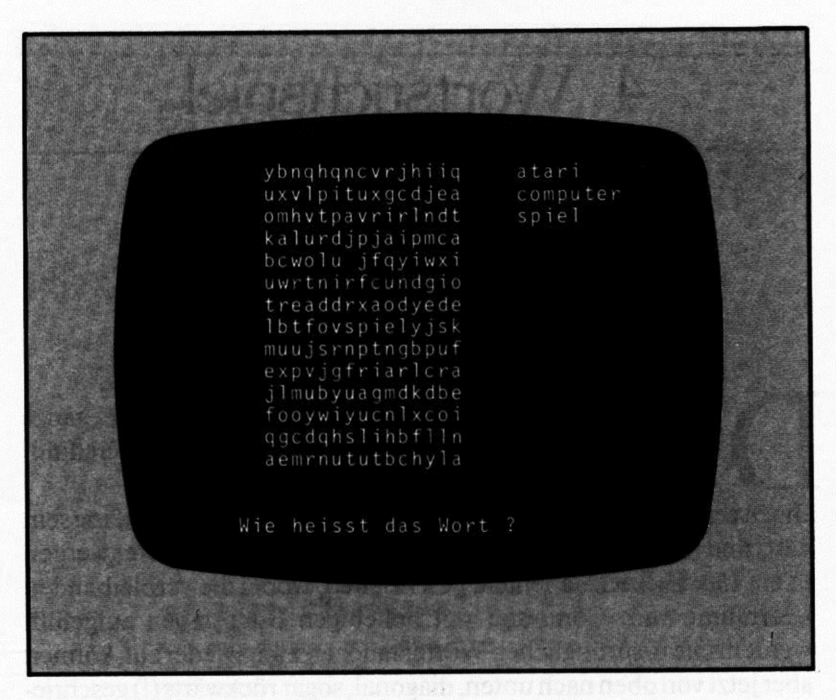
4 Wortsuchspiel

Dieses Puzzlespiel ist eine Herausforderung an die ganze Familie. Es setzt Scharfsinn, ein offenes Auge und Geduld voraus.
Bis zu zehn Wörter, von denen jedes bis zu zehn Buchstaben lang sein darf, sind in den Atari einzugeben. Die eingegebenen Wörter werden in ein 15 x 15-Buchstabenfeld geschrieben, wobei die verbleibenden Leerräume vom Computer mit beliebigen Buchstaben aufgefüllt werden. Die ursprünglichen Wörter tauchen zwar wieder auf, können aber jetzt von oben nach unten, diagonal, sogar rückwärts (!) geschrieben sein. Durch die Füllbuchstaben ist das Suchen der eingegebenen Wörter wie das Suchen der Nadel in dem berühmten Heuhaufen. Es erleichtert die Aufgabe, sich das Puzzle ohne die Füllbuchstaben anzuschauen, und man kann sich eine Liste der eingegebenen Wörter neben dem Puzzle ausgeben lassen. Aber auch dann ist es immer noch äußerst schwierig, die versteckten Wörter zu finden.

Ziel des Spiels ist es, alle versteckten Wörter zu finden. Für jedes wiedergefundene Wort gibt es einen Punkt.

SPIELVERLAUF

Die ersten Schritte des Spiels werden durch den Computer vorgegeben. Zunächst fordert er die Anzahl der Suchwörter an. Es müssen mehr als zwei Worte gewählt werden. Bei Falscheingabe erfolgt eine akustische Zurückweisung der Eingabe, und das Spiel beginnt von



vorn. Atari bittet nun um die Eingabe der zu versteckenden Wörter. Diese sind ausschließlich in Kleinbuchstaben einzugeben (auch der erste Buchstabe!). Für den anschließenden Aufbau des Puzzles wird eine gewisse Zeit benötigt; bitte nicht ungeduldig werden. Sicherheitshalber steht eine entsprechende Meldung auf dem Bildschirm. Die Rechenzeit hängt von der Anzahl und insbesondere von den Wortlängen der Suchwörter ab.

Im nächsten Schritt stehen die beiden Hilfsmöglichkeiten zur Wahl: Puzzle ohne Füllbuchstaben und Wörterliste neben dem Spielfeld. Werden beide Hilfen abgelehnt (Eingabe: «n»), startet das Spiel in seiner normalen, aber auch schwierigsten Version. Zunächst wird das Puzzle gezeichnet. Links oben befindet sich der Cursor, der wie üblich mit seinen Steuertasten bewegt werden kann. Wenn man meint, ein Wort gefunden zu haben, ist der Cursor auf den Anfangs-

buchstaben zu stellen und «w» einzugeben. Atari fordert nun das Lösungswort an. War es richtig, wird das Punktekonto (Score) um einen Punkt erhöht. Sind alle Begriffe gefunden worden, belohnt Atari diesen Erfolg durch einen Glückwunsch. Durch Eingabe von «r» während des Spiels gibt man auf und kehrt zu den Wahlmöglichkeiten der Hilfestellungen zurück.

HINWEISE

Bei diesem Spiel müssen alle Eingaben ausschließlich in Kleinbuchstaben erfolgen. Zwischen den Anführungszeichen in den Zeilen 410, 600, 800, 810, 930, 1400 und 1420 muß genau ein Leerzeichen stehen. Keine «Nullstrings» eingeben! Lediglich in Zeile 1090 existiert ein Nullstring: A$="".

PROGRAMMAUFBAU

15 Anfangswerte, Variablendefinitionen
80 Aufruf des Hauptprogramms
90 Eingabe der Ratewörter
280 Fehlermeldung bei zu langen Ratewörtern
330 Suche des längsten der eingegebenen Ratewörter
380 Aufbau des Puzzles
760 Ausgabe des Puzzles
870 Erlaubt, sich das Puzzle noch ohne die Füllbuchstaben anzusehen, setzt anschließend die Füllbuchstaben ein
1010 Hauptprogramm
1240 Eingabe eines vermeintlich gefundenen Ratewortes
1320 Verarbeitung eines richtig gefundenen Ratewortes
1500 Überprüfung des eingegebenen Ratewortes
1650 Setzt Punktestand auf Null
1670 Ausgabe des Puzzles ohne Füllbuchstaben
1710 Spielende
2000 Tonerzeugung

ERLÄUTERUNGEN

In diesem Spiel werden einige interessante Programmiertechniken vorgestellt. Für jedes Ratewort wird über die Random-Funktion ein zufälliger Startpunkt (Zeile 510 bis 520) und eine zufällige Richtung (Zeilen 530 bis 540) auf dem Spielfeld gewählt. Nun wird für jeden Buchstaben des Wortes geprüft, ob das für ihn vorgesehene Feld leer ist oder den Buchstaben schon enthält (von einem anderen Ratewort). Wenn das Wort paßt, wird es in die vorgesehenen Felder des Puzzles eingeschrieben. Diese Technik erlaubt es, die Wörter wie in einem Kreuzworträtsel anzuordnen. Paßt ein Wort nicht, werden ein neuer Startpunkt und eine neue Richtung gewählt, und das Prüfverfahren beginnt von neuem. So werden alle Wörter nach und nach in das Puzzle eingearbeitet.

Eines der Hauptprobleme von Atari-BASIC ist das Fehlen von Stringfeldern. Der einfachste Weg aus diesem Dilemma ist das «Pakken» der Werte. Insgesamt finden auf dem Spielfeld 15 x 15 = 225 Buchstaben Platz. Diese Buchstaben werden in dem String D$ gespeichert, der in Zeile 390 in der richtigen Länge dimensioniert worden ist. Damit kann das 15 x 15-Buchstabenfeld wie ein zweidimensionales Feld gespeichert werden. So steht der Buchstabe I,J (Zeile und Spalte) des Spielfelds in D$(I + (J-1) * 15, I + (J-1) * 15). Beispiel: Für I=5 und J=10 ergibt sich D$(150,150).

EIGENE ERWEITERUNGEN

Wenn ein Drucker zur Verfügung steht, sollte das Programm so ergänzt werden, daß das Puzzle ausgedruckt werden kann. Noch schwerer wird das Spiel, wenn Wörter mit mehr als zehn Buchstaben zugelassen werden. Sehr praktisch wäre es, wenn die gefundenen Wörter aus der Liste gestrichen oder sonst irgendwie kenntlich gemacht würden (z.B. im Puzzle selbst!).

```
10 REM Wortsuchspiel


15 OPEN #2,4,0,"K:"
20 DIM W$(30)
29 DIM LZ$(10)
30 LZ$="          ":REM 10 Leerzeichen
40 GOSUB 90
50 GOSUB 380
70 GOSUB 870

80 GOSUB 1010


90 GRAPHICS 0
100 DIM L$(100)
105 FOR I=1 TO 100:L$(I)=" ":NEXT I
110 DIM C(10)
120 LST=0
130 PRINT CHR$(125)
140 POSITION 5,2
150 POSITION 13,1:PRINT "Wortsuchspiel"
151 PRINT :PRINT "Bis zu 10 Worte sind in den Atari ein-";
152 PRINT "zugeben, die in einem Puzzle versteckt werden."
153 PRINT "Viel Erfolg beim Suchen!!!"
154 PRINT :PRINT "Wie viele Worte ";:INPUT W
155 PRINT
160 IF W<2 OR W>10 THEN GOSUB 2000:GOTO 140
170 PRINT "Rateworte in Kleinbuchstaben eingeben"
180 FOR I=1 TO W
190 POSITION 5,10+I:PRINT I;". Wort: ";
200 INPUT W$
210 IF LEN(W$)>10 THEN GOTO 280
220 IF LEN(W$)<1 THEN GOTO 200
230 L$((I-1)*10+1,(I-1)*10+LEN(W$))=W$
240 C(I)=LEN(W$)
250 NEXT I
270 RETURN


280 POSITION 1,22:PRINT "Nicht mehr als 10 Buchstaben!"
290 GOSUB 2000
292 POSITION 1,22:PRINT LZ$;LZ$;LZ$
295 POSITION 14,10+I:PRINT LZ$;LZ$;LZ$
300 GOTO 190
```

```
330 M=0:J=0
340 FOR Z=1 TO W
350 IF M<C(Z) THEN M=C(Z):J=Z
360 NEXT Z
370 RETURN

380 REM Worte ins Spielfeld setzen
390 DIM D$(225)
400 FOR J=1 TO 225
410 D$(J)=" "
420 NEXT J
450 PRINT CHR$(125)
460 F=0
470 FOR I=1 TO W
480 GOSUB 330
490 L=C(J)
500 C(J)=0
510 X=INT(RND(0)*(15-L))+1
520 Y=INT(RND(0)*(15-L))+1
530 V=INT(RND(0)*3)-1
540 U=INT(RND(0)*3)-1
550 IF U=0 AND V=0 THEN GOTO 530
560 A=X:B=Y
570 IF V<0 THEN A=A+L
580 IF U<0 THEN B=B+L
590 FOR K=1 TO L
595 H=(J-1)*10+K
596 W$(1,1)=L$(H,H)
600 H=A+(B-1)*15
601 IF D$(H,H)<>" " AND D$(H,H)<>W$(1,1) THEN F=1:K=L:GOTO 630
610 A=A+V
620 B=B+U
630 NEXT K
640 IF F=1 THEN F=0:GOTO 510
650 PRINT "Gleich geht es weiter ..."
660 A=X:B=Y
670 IF V<0 THEN A=A+L
680 IF U<0 THEN B=B+L
690 FOR K=1 TO L
700 H=A+(B-1)*15:G=(J-1)*10+K
701 D$(H,H)=L$(G,G)
710 A=A+V
720 B=B+U
730 NEXT K
740 NEXT I
750 RETURN
```

```
760 REM Puzzle auf Bildschirm
765 POKE 752,1
770 PRINT CHR$(125)
780 FOR M=1 TO 15
790 FOR N=1 TO 15
800 H=(M-1)*15+N
801 IF D$(H,H)=" " THEN PRINT ".";
810 IF D$(H,H)<>" " THEN PRINT D$(H,H);
820 NEXT N
830 IF LST=0 OR M>10 THEN PRINT
840 IF LST=1 AND M<=10 THEN POSITION 20,M:PRINT L$((M-1)*10+1,M*10)
850 NEXT M
860 RETURN

870 REM Puzzle ohne Fuellbuchstaben
880 DIM A$(1)
890 PRINT "Puzzle erst einmal ohne die Fuellbuch-";
891 PRINT "staben ausgeben j/n ";:INPUT A$
900 IF A$="j" THEN GOSUB 760
910 FOR I=1 TO 15
920 FOR J=1 TO 15
930 H=I+(J-1)*15
931 IF D$(H,H)=" " THEN D$(H,H)=CHR$(INT(RND(0)*25)+97)
940 NEXT J
950 NEXT I
960 PRINT "Sollen die Rateworte neben dem Puzzle"
961 PRINT "ausgegeben werden j/n ";:INPUT A$
970 IF LEN(A$)=0 THEN GOTO 960
980 IF A$="j" THEN LST=1
990 GOSUB 760
1000 RETURN


1010 X=2
1020 Y=1
1050 LOCATE X,Y,Q
1055 POSITION X,Y:PRINT CHR$(Q+128);
1080 GET #2,A
1090 IF A$="" THEN GOTO 1080
1100 IF A=119 THEN GOTO 1230
1110 POSITION X,Y:PRINT CHR$(Q);
1120 IF A=43 AND X>2 THEN X=X-1
1130 IF A=42 AND X<16 THEN X=X+1
1140 IF A=45 AND Y>1 THEN Y=Y-1
1150 IF A=61 AND Y<15 THEN Y=Y+1
1160 IF A=114 THEN GOSUB 1670
1220 GOTO 1050
```

```
1230 POSITION 0,20
1235 W$="          ":REM 10 Leerzeichen
1240 PRINT "Wie heisst das Wort ";:INPUT W$
1250 IF LEN(W$)=0 OR LEN(W$)>15 THEN GOSUB 2000:GOTO 1230
1260 GOSUB 1450
1270 POSITION 0,20:PRINT LZ$;LZ$;LZ$;LZ$
1280 IF MATCH=0 THEN GOSUB 2000:GOTO 1080
1290 FOR U=-1 TO 1
1300 FOR V=-1 TO 1
1310 IF U=0 AND V=0 THEN GOTO 1340

1320 GOSUB 1500
1330 IF MATCH=1 THEN V=1:U=1:GOTO 1340
1340 NEXT V
1350 NEXT U
1360 IF MATCH=1 THEN GOTO 1390
1370 GOSUB 2000
1380 GOTO 1080
1390 SCORE=SCORE+1
1400 POSITION 10,18:PRINT "Punktestand = ";SCORE;" "
1410 GOSUB 2000
1420 L$(10*(WORD-1)+1,10*(WORD-1)+1)=" "
1430 IF SCORE=W THEN GOTO 1780
1440 GOTO 1080
1450 MATCH=0
1460 FOR I=1 TO W
1470 H=(I-1)*10
1471 IF W$=L$(1+H,LEN(W$)+H) THEN MATCH=1:WORD=I:I=W
1480 NEXT I
1490 RETURN


1500 REM Wort gefunden ?
1510 MATCH=1
1520 A=X-1
1530 B=Y
1540 FOR I=1 TO LEN(W$)
1550 H=A+(B-1)*15
1551 IF W$(I,I)<>D$(H,H) THEN I=LEN(W$):MATCH=0:GOTO 1600
1560 A=A+U
1570 B=B+V
1580 IF A<1 OR A>15 THEN I=LEN(W$):MATCH=0:GOTO 1600
1590 IF B<1 OR B>15 THEN I=LEN(W$):MATCH=0:GOTO 1600
1600 NEXT I
1610 RETURN
.

1650 SCORE=0
1660 RETURN
```

```
1670 POSITION 0,19
1680 PRINT "Spiel wirklich abbrechen j/n ";:INPUT A$
1690 POSITION 0,19
1695 FOR I=0 TO 180:PRINT " ";:NEXT I
1700 IF A$<>"j" THEN RETURN


1710 PRINT CHR$(125)
1720 POSITION 10,8:PRINT "Ergebnis = ";SCORE
1730 POSITION 8,10
1740 PRINT "Noch ein Spiel j/n ";:INPUT A$
1750 IF A$="j" THEN RUN
1760 IF A$<>"n" THEN GOTO 1730
1770 STOP
1780 POSITION 0,19:PRINT "Toll, alle Worte gefunden !!!"
1790 GOTO 1740

2000 SOUND 0,80,10,8
2010 FOR T=1 TO 50
2020 NEXT T
2030 SOUND 0,0,0,0
2040 RETURN
```

[illegible]

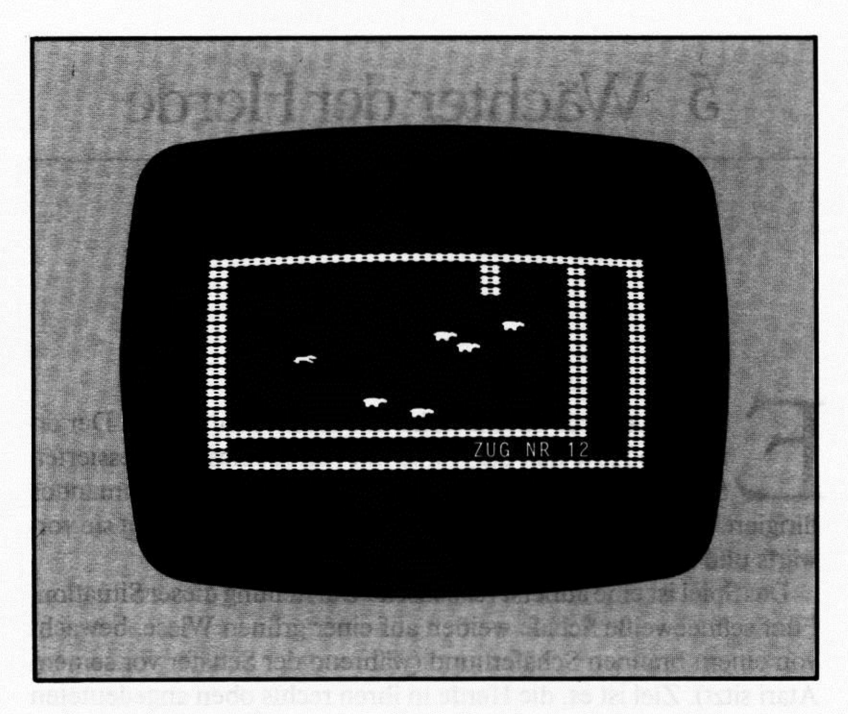
5 Wächter der Herde

Eine Schafherde strahlt Ruhe und Zufriedenheit aus. Der erfahrene Hirte hält die Herde mit Hilfe seines gut dressierten Schäferhundes zusammen, den er mit knappen Kommandos dirigiert. Der Hund umkreist unermüdlich die Herde, treibt sie vorwärts und spürt Ausreißer auf.

Das Spiel ist eine äußerst realistische Darstellung dieser Situation. Fünf schneeweiße Schafe weiden auf einer grünen Wiese, bewacht von einem braunen Schäferhund (während der Schäfer vor seinem Atari sitzt). Ziel ist es, die Herde in ihren rechts oben angedeuteten Stall zu treiben. Bisher hat es nur einer geschafft, vom Atari mit dem Titel eines Superhirten ausgezeichnet zu werden, und das war ein richtiger Schäfer ...

SPIELVERLAUF

Aufgabe des Hirten ist es, alle Schafe mit möglichst wenigen Spielzügen in den Stall in der rechten oberen Ecke des Bildschirms zu treiben. Die Kommandos für den Schäferhund werden dabei über die Pfeiltasten gegeben. Wenn der Hund jedoch den Schafen zu nahe kommt, zerstreuen sie sich in alle Himmelsrichtungen. Auch während des Spiels kann es plötzlich dazu kommen, daß die Schafe auseinanderlaufen. Für diesen Effekt wird die RND-Funktion des Atari ausgenutzt. Der Hund kann die Schafe durch unvorsichtiges Annähern so erschrecken, daß sie sogar über den Zaun springen. Wenn sich



die Schafe sehr eng zusammendrängen, kann man sie auf dem Bildschirm unter Umständen nicht mehr unterscheiden, so daß im Extremfall nur ein Schaf zu sehen ist.

Bei diesem Spiel wird man schnell gute und schlechte Strategien herausfinden. Es kostet sehr viel Zeit, den Hund um die Herde kreisen zu lassen. Um jedoch ein Lob des Atari zu erhalten, muß das Ziel mit möglichst wenigen Spielzügen erreicht werden.

HINWEISE

Der Zaun um die Wiese wird mit Hilfe des Doppelkreuzes (# = SHIFT-3) dargestellt (Unterprogramm ab Zeile 20). Zum Löschen der alten Positionen von Hund und Schafen wird das zwischen den Anführungsstrichen eingeschlossene Leerzeichen verwendet (Zeilen 440 und 780).

PROGRAMMAUFBAU

20	Initialisierung der Grafik und der Variablen
120	Umzäunung der Wiese
240	Stall
280	Darstellung eines Schafes
340	Darstellung des Hundes
390	Bewegung des Hundes
520	Bewegung der Schafe und Test auf Spielende
550	Berechnungen für die Bewegungen der Schafe
780	Ausgabe der Schafe auf dem Bildschirm
860	Meldung über den Spielstand; Spielende
990	Auseinanderlaufen der Schafe
2000	Definition des Grafikzeichens für ein Schaf
2100	Definition des Grafikzeichens für den Hund
2200	Umschaltung des Atari auf den neu definierten RAM-Zeichensatz

PROGRAMMHINWEISE

Die Tastatur wird mit Hilfe der GET-Anweisung gelesen. Vorteil hierbei ist, daß jeder Tastendruck sofort der GET-Variablen zugewiesen wird, ohne daß die Eingabe mit RETURN abgeschlossen werden muß. Die Aktionen des Hundes können so wesentlich einfacher gesteuert werden. Für die GET-Anweisung muß die Tastatur als Eingabeeinheit bestimmt werden, was in Zeile 30 durch die OPEN-Anweisung geschieht. Die GET-Anweisung unterbricht das Programm, wartet auf einen Tastendruck, weist den Tastencode der GET-Variablen zu und setzt das Programm mit der nächsten Anweisung fort.

Trotz der vorbildgerechten Darstellung des Herdenlebens sind für die einzelnen Tierbewegungen keine speziellen Tricks erforderlich. Die Berechnungen der gegenseitigen Entfernungen erfolgen durch mathematische Gleichungen: In den Zeilen 600 und 610 wird z.B. geprüft, ob sich der Hund den Schafen zu sehr genähert hat. Falls ja, wird das Zerstreuen der Schafe durch die Gleichungen in den Zeilen 1000 und 1030 berechnet. Das Zerstreuen der Schafe ohne erkennbaren

Grund wird über eine Zufallszahl gesteuert, wobei nur Zahlen unter 0.1 das Auseinanderlaufen bewirken (Wahrscheinlichkeit hierfür also 1 : 100). IF-Anweisungen stellen sicher, daß Schafe weder auf dem Zaun noch auf dem Hund sitzen können.

EIGENE ERWEITERUNGEN

Der Reiz des Spiels kann durch Hindernisse wie Bäume, ein Fluß mit oder ohne Brücke noch gesteigert werden. Durch Ändern des Schwellenwerts kann das Zerstreuen der Schafe häufiger erfolgen. Man kann auch gegen die Uhr spielen, wie es im Spiel «Achtung – fertig – los!» vorgesehen ist.

```
10 REM Waechter der Herde


20 GRAPHICS 1+16
25 PRINT #6;" WAECHTER DER HERDE"
26 POSITION 3,16
27 PRINT #6;"es geht gleich"
28 POSITION 7,19
29 PRINT #6;"weiter"
30 OPEN #2,4,0,"K:"
40 CH=(PEEK(106)-8)*256
50 CHORG=(PEEK(756)*256)
60 FOR I=0 TO 511
70 POKE CH+I,PEEK(CHORG+I)
80 NEXT I
90 GOSUB 2000
100 DIM Y(5):DIM X(5)
110 M=0


120 FOR X=0 TO 15
130 POSITION X,16:PRINT #6;"#"
140 NEXT X
150 FOR Y=0 TO 16
160 POSITION 16,Y:PRINT #6;"#"
170 NEXT Y
180 FOR Y=0 TO 19
190 POSITION 0,Y:PRINT #6;"#";
191 POSITION 19,Y:PRINT #6;"#"
200 NEXT Y
210 FOR X=0 TO 19
220 POSITION X,0:PRINT #6;"#";
221 POSITION X,19:PRINT #6;"#"
230 NEXT X
240 FOR Y=1 TO 3
250 POSITION 12,Y:PRINT #6;"#"
260 NEXT Y

280 FOR S=1 TO 5
290 Y(S)=5+INT(RND(0)*10)
300 X(S)=4+INT(RND(0)*6)
310 POSITION X(S),Y(S):PRINT #6;CHR$(4);
320 NEXT S

340 YD=1+INT(RND(0)^3)
350 XD=1+INT(RND(0)^3)
360 POSITION XD,YD:PRINT #6;CHR$(5+160)
```

```
390 GET #2,D
400 IF D=0 THEN GOTO 390
410 IF XD=12 AND YD=4 AND D=45 THEN GOTO 390
420 IF XD=11 AND YD<4 AND D=42 THEN GOTO 390
430 IF XD=13 AND YD<4 AND D=43 THEN GOTO 390
440 POSITION XD,YD:PRINT #6;" "
450 IF D=43 AND XD>1 THEN XD=XD-1
460 IF D=42 AND XD<15 THEN XD=XD+1
470 IF D=61 AND YD<15 THEN YD=YD+1
480 IF D=45 AND YD>1 THEN YD=YD-1
490 POSITION XD,YD:PRINT #6;CHR$(5+160)
500 M=M+1
510 POSITION 10,18
511 PRINT #6;"zug nr ";M

520 GOSUB 550
530 IF F=0 THEN GOTO 860
540 GOTO 390

550 F=0
570 FOR S=1 TO 5
580 Y=Y(S)
590 X=X(S)
600 HX=ABS(X(S)-XD):HY=ABS(Y(S)-YD)
601 IF (HX<2 AND HY<2) THEN GOSUB 990
610 IF RND(0)<0.01 THEN GOSUB 990
620 IF ABS(X(S)-XD)>2+RND(0)*2 THEN GOTO 780
630 IF ABS(Y(S)-YD)>2+RND(0)*2 THEN GOTO 780
640 X(S)=X(S)+SGN(X(S)-XD)
650 Y(S)=Y(S)+SGN(Y(S)-YD)
660 IF X(S)<13 AND X(S)>11 AND Y(S)<4 THEN X(S)=X
670 FS=0
680 FOR Z=1 TO 5
690 IF Z=S THEN GOTO 710
700 IF (X(S)=X(Z)) AND (Y(S)=Y(Z)) THEN FS=1
710 NEXT Z
720 IF FS=1 THEN GOTO 640
730 IF X(S)=XD AND Y(S)=YD THEN GOTO 640
740 IF X(S)<1 THEN X(S)=1
750 IF X(S)>15 THEN X(S)=15
760 IF Y(S)<1 THEN Y(S)=1
770 IF Y(S)>15 THEN Y(S)=15

780 POSITION X,Y:PRINT #6;" "
790 POSITION X(S),Y(S):PRINT #6;CHR$(4);
800 IF X(S)>12 AND (Y(S)>0 AND Y(S)<4) THEN GOTO 820
```

```
810 F=1
820 NEXT S
830 RETURN
860 POSITION 10,18:PRINT #6;"   ende   "
861 FOR I=1 TO 1000:NEXT I
870 IF M<40 THEN PRINT "Superhirte!":GOTO 920
880 IF M<60 THEN PRINT "Brauchbarer Hund!":GOTO 920
890 IF M<90 THEN PRINT "Weiter ueben!!":GOTO 920
900 IF M<120 THEN PRINT "es bessert sich ...":GOTO 920
910 PRINT "Das dauerte viel zu lange!"
920 POSITION 0,10:PRINT "Es wurden ";M;
921 PRINT " Spielzuege benoetigt"
930 DIM A$(1)
935 PRINT :PRINT :PRINT
940 PRINT "Noch ein Spiel J/N ";:INPUT A$
950 IF A$="J" THEN RUN
960 GRAPHICS 0
980 END

990 XT=X(S):YT=Y(S)
1000 X(S)=X(S)+(SGN(RND(0)-0.5)*(2+RND(0)*2))
1010 IF X(S)<1 THEN X(S)=1
1020 IF X(S)>15 THEN X(S)=15
1030 Y(S)=Y(S)+(SGN(RND(0)-0.5)*(2+RND(0)*2))
1035 Y(S)=INT(Y(S)):X(S)=INT(X(S))
1040 IF Y(S)<1 THEN Y(S)=1
1050 IF Y(S)>15 THEN Y(S)=15
1060 IF X(S)=12 AND Y(S)<3 THEN GOTO 1000
1070 IF XT=X(S) AND YT=Y(S) THEN GOTO 1000
1080 RETURN

2000 POKE CH+(ASC("$")-32)*8+0,0
2010 POKE CH+(ASC("$")-32)*8+1,0
2020 POKE CH+(ASC("$")-32)*8+2,122
2030 POKE CH+(ASC("$")-32)*8+3,255
2040 POKE CH+(ASC("$")-32)*8+4,125
2050 POKE CH+(ASC("$")-32)*8+5,120
2060 POKE CH+(ASC("$")-32)*8+6,72
2070 POKE CH+(ASC("$")-32)*8+7,72

2100 POKE CH+(ASC("%")-32)*8+0,0
2110 POKE CH+(ASC("%")-32)*8+1,0
2120 POKE CH+(ASC("%")-32)*8+2,6
2130 POKE CH+(ASC("%")-32)*8+3,123
2140 POKE CH+(ASC("%")-32)*8+4,120
```

```
2150 POKE CH+(ASC("%")-32)*8+5,132
2160 POKE CH+(ASC("%")-32)*8+6,66
2170 POKE CH+(ASC("%")-32)*8+7,0
2200 POKE 752,1
2210 POKE 756,CH/256

2220 SETCOLOR 0,0,0
2230 SETCOLOR 1,0,14
2240 SETCOLOR 2,2,5
2250 SETCOLOR 4,13,8
2500 RETURN
```

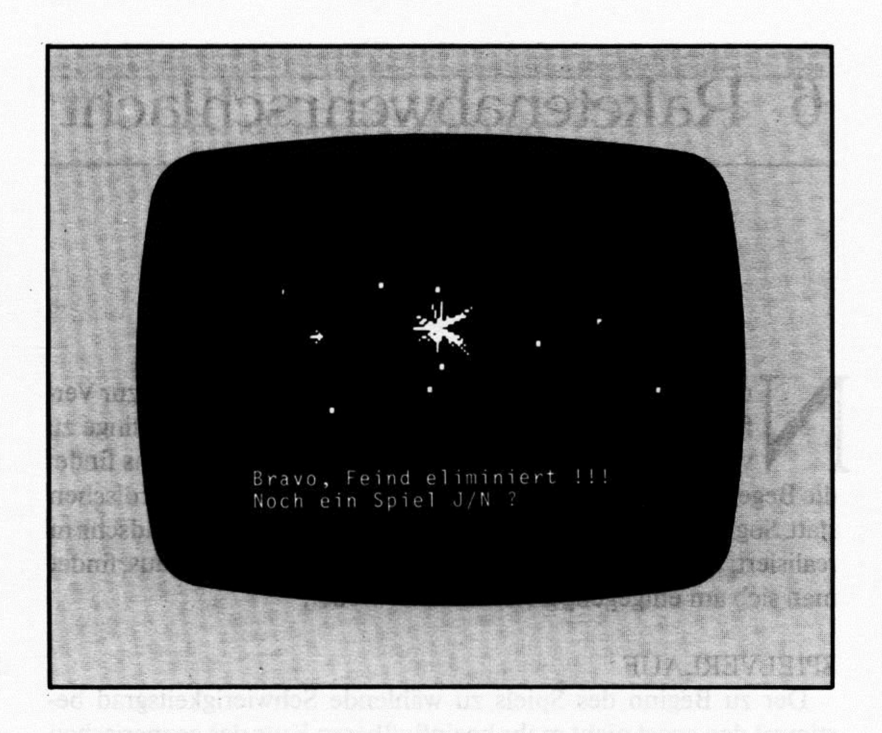
6 Raketenabwehrschlacht

Nur 100 Sekunden stehen in diesem aufregenden Spiel zur Verfügung, um ein Raumschiff außerirdischer Eindringlinge zu vernichten. In der unendlichen Weite des Weltraums findet die Begegnung eines terrestrischen Schiffes mit den Außerirdischen statt. Sogar die Einsteinsche Raumkrümmung ist auf dem Bildschirm realisiert: Fliegt man über einen Rand des Bildschirms hinaus, findet man sich am entgegengesetzten Rand wieder.

SPIELVERLAUF

Der zu Beginn des Spiels zu wählende Schwierigkeitsgrad bestimmt den sonst nicht mehr beeinflußbaren Kurs des gegnerischen Raumschiffes und die Anzahl der Sterne. Der Navigator muß das Schiff sorgfältig um die Sterne herumführen, beim Rammen eines Sternes wird das Raumschiff irgendwo in den Raum geschleudert (RND-Funktion). Das eigene terrestrische Raumschiff wird in Form eines Pfeils dargestellt und bewegt sich mit konstanter Geschwindigkeit vorwärts. Der Navigator kann das Raumschiff lediglich in die richtige Richtung dirigieren. Bei jedem Tastendruck wird das Raumschiff um 45 Grad im Uhrzeigersinn gedreht.

Das Abfeuern des Lasers erfolgt über die Leertaste. Der Laserstrahl zischt in der durch das eigenen Raumschiff vorgegebenen Richtung durch das All. Wird das feindliche Raumschiff getroffen, explodiert es geräuschvoll. Die laufend auf dem Schirm eingeblendete Spielzeit beträgt maximal 100 Sekunden.



HINWEISE

Beim Atari mit 16-k-RAM müssen vorhandene Diskettenlaufwerke abgeschaltet werden, damit für dieses sehr lange Programm genügend Speicherplatz zur Verfügung steht. Zum Abspeichern des Programms muß ein Kassettenrecorder verwendet werden.

PROGRAMMAUFBAU

20 Hauptprogramm
1000 Richtung des terrestrischen Raumschiffs bzw. Laserstrahls
2000 Laserstrahl
2235 Explosion des gegnerischen Raumschiffs
2500 Prüfung, ob Gegner getroffen worden ist
3000 Flug des eigenen Raumschiffs

4000 Flug des feindlichen Raumschiffs
5000 Löschen der alten Position des eigenen Raumschiffs
5100 Ausgabe des eigenen Raumschiffs auf dem Bildschirm
5200 Tonteil
5300 Ausgabe des gegnerischen Raumschiffs auf dem Bildschirm
5500 Ton für Laserstrahl
7000 Titelbild
7500 Zeitmessung und Anzeige auf dem Bildschirm
7900 Ausgabe der Sterne auf dem Bildschirm
8000 Anfangswerte der Variablen, Farben
9000 Ende des Spiels

ERLÄUTERUNGEN

Das Programm ist ziemlich kompliziert, enthält aber interessante Tricks. Das eigene pfeilförmige und das gegnerische radförmige Raumschiff werden mit Hilfe der PLOT- und DRAWTO-Anweisungen gezeichnet. In den Feldern V und W werden die horizontalen und vertikalen Geschwindigkeiten der beiden Raumschiffe gespeichert und für die Bildschirmpositionen ausgewertet. Aus diesen Werten wird auch die Bewegungsrichtung des eigenen Raumschiffs und dessen entsprechende Ausrichtung auf dem Bildschirm errechnet.

EIGENE ERWEITERUNGEN

Die Spannung des Spiels läßt sich noch um ein Vielfaches steigern. So könnte das gegnerische Raumschiff ebenfalls schießen, was eigene Ausweichmanöver zur Folge hätte. Zur Steuerung der feindlichen Kanonen wäre die RND-Funktion gut geeignet.

```
10 REM Raketenabwehrschlacht

20 GOSUB 7000
30 GOSUB 8000
40 GOSUB 7500
50 IF INT(T)>1000 THEN GOTO 9000
60 GOSUB 1000
70 GOSUB 3000
80 IF H=1 THEN GOTO 9000
90 GOSUB 4000
100 GOTO 40

1000 I=PEEK(764)
1010 IF I=255 THEN RETURN
1015 POKE 764,255
1020 IF I=33 THEN GOTO 2000
1030 K=K+1
1040 IF K>8 THEN K=1
1050 RETURN

2000 XL=X+4*V(K)
2010 YL=Y+4*W(K)
2015 GOSUB 2500
2050 PLOT XL,YL
2060 DX=0
2070 IF V(K)=1 THEN DX=144-XL
2080 IF V(K)=-1 THEN DX=6-XL
2090 DY=0
2100 IF W(K)=1 THEN DY=72-YL
2120 IF W(K)=-1 THEN DY=6-YL
2130 IF V(K)*W(K)=0 THEN GOTO 2200
2140 HH=ABS(DX)
2141 IF HH<ABS(DY) THEN DY=HH*SGN(DY):GOTO 2200
2150 DX=ABS(DY)*SGN(DX)
2200 COLOR 1:DRAWTO XL+DX,YL+DY
2210 COLOR 0:PLOT XL,YL
2215 GOSUB 5500
2220 COLOR 0:DRAWTO XL+DX,YL+DY
2230 IF H=0 THEN RETURN

2235 MX=B+2:MY=A-2
2240 FOR I=1 TO INT(RND(0)*5)+20
2250 COLOR 1:PLOT MX,MY
2260 DX=10-RND(0)*20
2270 IF MX+DX>159 OR MX+DX<0 THEN GOTO 2330
```

```
2280 DY=10-RND(0)*20
2290 IF MY+DY>79 OR MY+DY<0 THEN GOTO 2330
2300 PLOT MX,MY
2310 DRAWTO MX+DX,MY+DY
2320 SOUND 0,INT(RND(0)*100),4,8
2330 NEXT I
2340 RETURN


2500 H=0
2510 DY=A-Y-4*W(K)
2520 DX=B-X-4*V(K)
2530 IF W(K)*DX<>V(K)*DY THEN RETURN
2540 IF ABS(V(K))*SGN(DX)<>V(K) THEN RETURN
2541 IF ABS(W(K))*SGN(DY)<>W(K) THEN RETURN
2550 H=1
2560 RETURN


3000 IF NB1=0 THEN GOSUB 5000
3010 X=X+6*V(K)
3020 Y=Y+6*W(K)
3030 IF X<6 THEN X=144
3040 IF X>144 THEN X=6
3050 IF Y<6 THEN Y=72
3060 IF Y>72 THEN Y=6
3065 LOCATE X+V(K)*4,Y+W(K)*4,Q
3066 IF Q<>0 THEN GOSUB 5200:NB1=1:GOTO 1030
3070 GOSUB 5100
3080 NB1=0
3090 RETURN


4000 IF RND(0)>1.05-DF/20 THEN Z=Z+1
4010 IF Z>8 THEN Z=1
4020 IF NB2=0 THEN GOSUB 5300
4030 A=A+6*V(Z)
4040 B=B+6*W(Z)
4050 IF B<6 THEN B=144
4060 IF B>144 THEN B=6
4070 IF A<6 THEN A=72
4080 IF A>72 THEN A=6
4085 LOCATE B+2,A+2,Q
4086 IF Q<>0 THEN GOSUB 5200:NB2=1:Z=Z+1:GOTO 4010
4090 R=R+1:IF R>8 THEN R=1
4100 GOSUB 5400
4110 NB2=0
4120 RETURN
```

```
5000 COLOR 0
5010 K=K-1
5020 GOSUB 5110
5030 K=K+1
5040 GOTO 5110

5100 COLOR 1
5110 PLOT X,Y
5120 DRAWTO X+4*V(K),Y+4*W(K)
5130 I=K+3:I=I-INT(I/8)*8
5140 DRAWTO X+4*V(K)+2*V(I),Y+4*W(K)+2*W(I)
5150 PLOT X+4*V(K),Y+4*W(K)
5160 I=K+5:I=I-INT(I/8)*8
5170 DRAWTO X+4*V(K)+2*V(I),Y+4*W(K)+2*W(I)
5180 RETURN

5200 SOUND 0,80,10,10
5210 FOR S=1 TO 10
5220 NEXT S
5230 SOUND 0,0,0,0
5240 RETURN

5300 COLOR 0
5310 GOTO 5410
5400 COLOR 1
5410 PLOT B-2*V(R),A-2*W(R)
5420 DRAWTO B+V(R)*2,A+W(R)*2
5430 I=R+2:I=I-INT(I/8)*8
5440 PLOT B-V(I)*2,A-W(I)*2
5450 DRAWTO B+V(I)*2,A+W(I)*2
5460 RETURN

5500 FOR S=255 TO 0 STEP 20
5510 SOUND 0,S,10,10:SOUND 0,S,10,14
5520 NEXT S
5530 SOUND 0,0,0,0
5540 RETURN

7000 GRAPHICS 0
7040 POSITION 10,2:PRINT "Raketenabwehrschlacht"
7045 POSITION 5,5
7050 PRINT " Sie sind der Kommandant eines"
7060 POSITION 6,7:PRINT "Raumschiffs in vorgerueckter"
```

```
7070 POSITION 6,9:PRINT "Position zur Abwehr fremder"
7075 POSITION 14,11:PRINT "Eindringlinge"
7080 POSITION 1,13:PRINT "Der Gegner muss innerhalb der vorge-"
7090 POSITION 1,15:PRINT "gebenen Zeit (100 s) erledigt sein!"
7100 POSITION 2,22:PRINT "Schwierigkeitsgrad waehlen:"
7105 PRINT "<1> leicht ... <10> schwer ";
7110 INPUT DF
7120 IF DF<1 OR DF>10 THEN GOTO 7100
7130 RETURN


7500 T=(PEEK(20)+PEEK(19)*256)/5
7510 PRINT INT(T)/10;"  "
7520 RETURN


7900 REM Sterne
7910 XS=INT(RND(0)*159)
7920 YS=INT(RND(0)*60)
7930 COLOR 1
7940 PLOT XS,YS:PLOT XS+1,YS
7941 PLOT XS,YS+1:PLOT XS+1,YS+1
7950 RETURN

8000 DIM W(8),V(8),A$(1)
8010 GRAPHICS 7
8020 DATA -1,1,0,1,1,1,1,0,1,-1,0,-1,-1,-1,-1,0,-1,1
8030 FOR I=0 TO 8
8040 READ W,V:W(I)=W:V(I)=V
8050 NEXT I
8060 K=1
8070 X=3*6+6
8080 Y=4*6+6
8090 V=V(K)
8100 W=W(K)
8105 SETCOLOR 1,0,14:SETCOLOR 2,7,8
8106 SETCOLOR 4,7,8:SETCOLOR 0,0,14
8110 FOR J=1 TO 10+RND(0)*DF:GOSUB 7900:NEXT J
8120 POKE 18,0
8130 POKE 19,0
8140 POKE 20,0
8150 B=INT(RND(0)*10)*6+6
8160 A=INT(RND(0)*10)*6+6
8900 RETURN
```

```
9000 IF H<>1 THEN GOTO 9500
9010 PRINT "Bravo, Feind eliminiert!"
9020 GOTO 9600
9500 PRINT "Schade, die Zeit ist um!"
9600 SOUND 0,0,0,0
9610 PRINT "Noch ein Spiel J/N ";:INPUT A$
9620 IF A$="J" THEN RUN
```



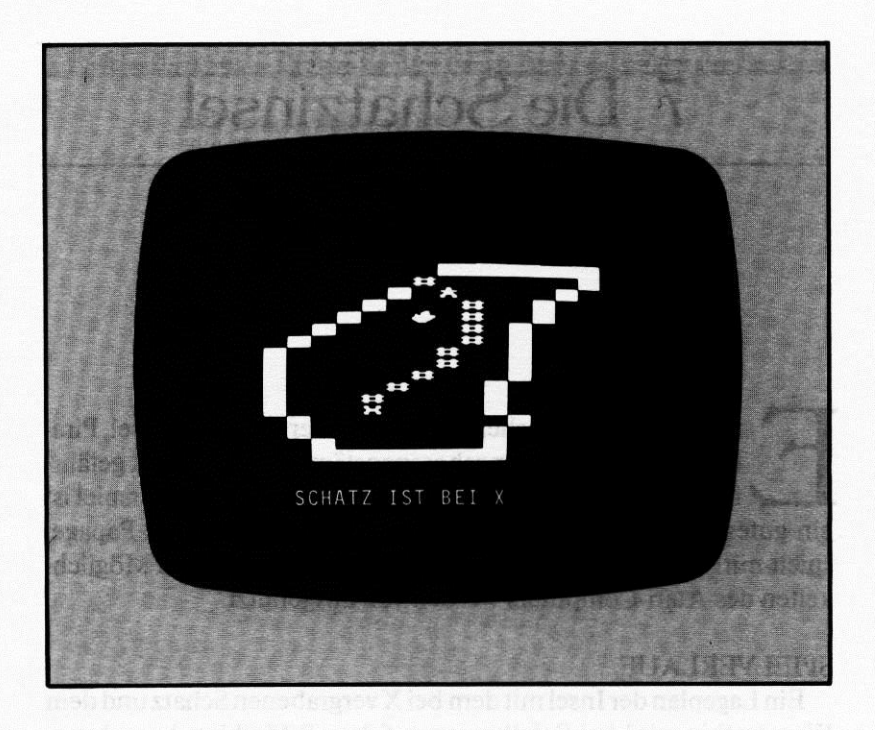
7 Die Schatzinsel

Ein unermeßlich reicher Schatz auf einer einsamen Insel, Piraten, verschiedene Eingeborenenstämme, Stellen mit gefährfährlichem Treibsand. – In diesem echten Abenteuerspiel ist ein gutes Gedächtnis Gold wert! Sogar Long-John-Silvers Papagei spielt mit! Die umfangreichen grafischen und akustischen Möglichkeiten des Atari-Computers werden voll ausgenutzt.

SPIELVERLAUF

Ein Lageplan der Insel mit dem bei X vergrabenen Schatz und dem Weg dorthin wird bei Spielbeginn auf dem Bildschirm kurz dargestellt. Man sollte sich das Bild gut einprägen, denn die Landkarte verschwindet relativ schnell wieder (RND-Funktion). Ein Verlassen des Weges kann fatale Folgen haben. Ein Versinken im Treibsand beendet das Spiel unter Umständen ziemlich rasch, es sei denn, ein befreundeter Eingeborener taucht als Retter auf. Unangenehm ist es, feindlichen Ureinwohnern der Insel zu begegnen, da man durch die langen Verhandlungen wieder ein Stück zurückgeworfen wird. Long-John-Silvers Papagei ist ein echter Kumpel: er fliegt genau über dem nächsten Wegstück und dient so als lebendiger Wegweiser. Leider ist dieser Vogel viel zu selten in der Nähe!

Während der Schatzsuche nähert sich ein Piratenschiff der Insel. Wenn die Piraten landen können, ist das Spiel zu Ende. Jede Verzögerung sollte also vermieden werden. Die Geschwindigkeit, mit der sich



das Schiff nähert, hängt von den nachfolgend beschriebenen Bedingungen ab.

Die Karte wird während des Spiels durch Eingabe von «H» erneut kurz eingeblendet. Da auch alle Hindernisse gezeigt werden, ist das schon eine große Hilfe. Allerdings kommt als «Strafe» das Piratenschiff ein großes Stück näher, so daß von dieser Hilfsmöglichkeit nur selten Gebrauch gemacht werden sollte.

Nach je fünf Spielzügen wird die Insel mit dem Piratenschiff gezeigt, die Entfernung nimmt dabei stetig ab. Für die Schatzsuche werden die Pfeiltasten nach rechts, links und unten benötigt. Der Weg zurück ist nicht möglich. Viel Erfolg!

HINWEISE

Bei Ataris mit 16-k-RAM müssen die Diskettenlaufwerke abgeschaltet werden, um genügend Speicherplatz zu erhalten. Zum Speichern des Programms muß ein Kassettenrecorder verwendet werden.

PROGRAMMAUFBAU

20 Kopiert Zeichensatz in den RAM-Bereich
100 Darstellung eines Männchens im RAM-Zeichensatz
200 Darstellung des Treibsands im RAM-Zeichensatz
300 Darstellung des Papageis im RAM-Zeichensatz
400 Darstellung eines Eingeborenen im RAM-Zeichensatz
500 Darstellung eines Teils des Piratenschiffs im RAM-Zeichensatz
600 Darstellung eines Teils des Piratenschiffs im RAM-Zeichensatz
700 Blockgrafik für den Rand der Insel
800 Umschalten auf den RAM-Zeichensatz und Abschalten des Cursors
820 Initialisierung der Variablen und Felder
1000 Hauptprogramm
1200 Test, ob Schatzsucher in Treibsand geraten ist
1300 Angriff feindlicher Eingeborener
2000 Programmlogik für den Hilfsflug des Papageis
3000 Ausgabe der Landkarte auf dem Bildschirm
3500 Bewegung des Schatzsuchers
3700 Ausgabe des Treibsands auf der Landkarte
4000 Berechnung, wo in diesem Spiel Treibsandflächen sein sollen
4500 Tonerzeugung
4600 Löscht den Bildschirm und schaltet um auf Grafik
5000 Form der Insel
5500 Weg zu dem Schatz
6000 Ausgabe und Fahrt des Piratenschiffs
7000 Ausgabe der Insel auf dem Bildschirm

8000 Hilfestellung
9000 Schatz gefunden
9990 Ende des Spiels

ERLÄUTERUNGEN

Das Programm ist trotz seiner Länge, seiner vielen Feinheiten und bemerkenswerten Techniken übersichtlich und vor allen Dingen gradlinig aufgebaut. Das Unterprogramm ab Zeile 5000, das die Form der Insel berechnet, verdient besondere Aufmerksamkeit. Hier sind die Vorzeichenfunktion (SGN) und der Zufallszahlengenerator (RND) kombiniert worden, um die charakteristische unregelmäßige Form einer Insel als geschlossene Linie in Blockgrafik darzustellen.

EIGENE ERWEITERUNGEN

Die Zeit, während der die Karte auf dem Bildschirm sichtbar ist, kann durch Ändern der Parameter in der Warteschleife ab Zeile 8020 variiert werden. Die Geschwindigkeit des Piratenschiffs wird durch den Wert der Variablen R (Zeile 6020) bestimmt, die während des Spiels Werte zwischen 0 und 2 annimmt (RND-Funktion). Änderungen dieses Wertes haben direkten Einfluß auf den Spielablauf.

```
10 REM Die Schatzinsel

20 GRAPHICS 1+16
21 POSITION 9,0:PRINT #6;"die"
22 POSITION 5,10
23 PRINT #6;"schatzinsel"
24 POSITION 0,19
25 PRINT #6;"GLEICH GEHT´S WEITER"
30 OPEN #2,4,0,"K:"
40 CH=(PEEK(106)-8)*256
50 CHORG=(PEEK(756)*256)
60 FOR I=0 TO 511
70 POKE CH+I,PEEK(CHORG+I)
80 NEXT I

100 POKE CH+(ASC("$")-32)*8+0,24
110 POKE CH+(ASC("$")-32)*8+1,24
120 POKE CH+(ASC("$")-32)*8+2,126
130 POKE CH+(ASC("$")-32)*8+3,24
140 POKE CH+(ASC("$")-32)*8+4,60
150 POKE CH+(ASC("$")-32)*8+5,102
160 POKE CH+(ASC("$")-32)*8+6,102
170 POKE CH+(ASC("$")-32)*8+7,0

200 POKE CH+(ASC("@")-32)*8+0,12
210 POKE CH+(ASC("@")-32)*8+1,28
220 POKE CH+(ASC("@")-32)*8+2,92
230 POKE CH+(ASC("@")-32)*8+3,127
240 POKE CH+(ASC("@")-32)*8+4,255
250 POKE CH+(ASC("@")-32)*8+5,254
260 POKE CH+(ASC("@")-32)*8+6,124
270 POKE CH+(ASC("@")-32)*8+7,56

300 POKE CH+(ASC("&")-32)*8+0,32
310 POKE CH+(ASC("&")-32)*8+1,54
320 POKE CH+(ASC("&")-32)*8+2,62
330 POKE CH+(ASC("&")-32)*8+3,28
340 POKE CH+(ASC("&")-32)*8+4,30
350 POKE CH+(ASC("&")-32)*8+5,39
360 POKE CH+(ASC("&")-32)*8+6,64
370 POKE CH+(ASC("&")-32)*8+7,0
```

```
400 POKE CH+(ASC("%")-32)*8+0,1
410 POKE CH+(ASC("%")-32)*8+1,25
420 POKE CH+(ASC("%")-32)*8+2,217
430 POKE CH+(ASC("%")-32)*8+3,255
440 POKE CH+(ASC("%")-32)*8+4,217
450 POKE CH+(ASC("%")-32)*8+5,25
460 POKE CH+(ASC("%")-32)*8+6,37
470 POKE CH+(ASC("%")-32)*8+7,37

500 POKE CH+(ASC("[")-32)*8+0,255
510 POKE CH+(ASC("[")-32)*8+1,126
520 POKE CH+(ASC("[")-32)*8+2,165
530 POKE CH+(ASC("[")-32)*8+3,195
540 POKE CH+(ASC("[")-32)*8+4,195
550 POKE CH+(ASC("[")-32)*8+5,165
560 POKE CH+(ASC("[")-32)*8+6,126
570 POKE CH+(ASC("[")-32)*8+7,255

600 POKE CH+(ASC("]")-32)*8+0,16
610 POKE CH+(ASC("]")-32)*8+1,255
620 POKE CH+(ASC("]")-32)*8+2,255
630 POKE CH+(ASC("]")-32)*8+3,126
640 POKE CH+(ASC("]")-32)*8+4,126
650 POKE CH+(ASC("]")-32)*8+5,126
660 POKE CH+(ASC("]")-32)*8+6,60
670 POKE CH+(ASC("]")-32)*8+7,60

700 POKE CH+(ASC("´")-32)*8+0,255
710 POKE CH+(ASC("´")-32)*8+1,255
720 POKE CH+(ASC("´")-32)*8+2,255
730 POKE CH+(ASC("´")-32)*8+3,255
740 POKE CH+(ASC("´")-32)*8+4,255
750 POKE CH+(ASC("´")-32)*8+5,255
760 POKE CH+(ASC("´")-32)*8+6,255
770 POKE CH+(ASC("´")-32)*8+7,255


800 POKE 756,CH/256
810 POKE 752,1


820 F=0
830 XS=1:YS=1
840 MES=0
850 DIM V(11)
860 DIM U(11)
```

```
870 DIM L(21)
880 DIM R(21)
890 DIM X(21)
900 DIM A$(10)

1000 GOSUB 5000
1010 XM=X(T+1)
1070 YM=T+1
1080 GOSUB 3000
1090 FOR Q=1 TO 300+INT(RND(0)*200):NEXT Q
1100 GOSUB 7000
1110 GOSUB 3600
1120 GOSUB 3500
1130 IF XM=XT AND YM=YT THEN GOTO 9000
1140 F=F+1
1150 IF INT(F/5)=F/5 THEN GOSUB 6000
1160 IF X(YM)=XM AND INT(F/5)=F/5 THEN GOTO 1100
1170 IF X(YM)=XM THEN GOTO 1120

1200 GOSUB 4500
1210 GOSUB 6000
1220 FOR Q=1 TO 10
1230 IF V(Q)=XM AND U(Q)=YM THEN GOSUB 4000
1240 NEXT Q
1250 IF RND(0)<=0.4 THEN GOTO 2000

1300 GOSUB 7000
1310 POSITION 1,19
1320 PRINT #6;"achtung feindliche"
1325 PRINT #6;" eingeborene"
1330 FOR N=1 TO 3
1340 R=INT(RND(0)*3)
1350 IF YM+R>=B THEN R=0
1360 POSITION XM,YM+R
1370 PRINT #6;"%"
1380 NEXT N
1390 YM=YM-3
1400 IF YM<=T+1 THEN YM=T+1
1410 XM=X(YM)
1420 MES=1
1430 POSITION XM,YM
1440 PRINT #6;"$";
1450 GOTO 1120
```

```
2000 GOSUB 7000
2010 GOSUB 3600
2020 POSITION 1,19
2030 PRINT #6;"dem papagei nach"
2040 YJ=YM+1
2050 IF YJ>P THEN YJ=P
2060 XJ=X(YJ)
2070 POSITION XJ,YJ
2080 PRINT #6;CHR$(6)
2090 MES=1
2100 GOTO 1120

3000 GOSUB 4600
3020 FOR X=L(T) TO R(T)
3030 POSITION X,T
3040 PRINT #6;CHR$(7)
3050 NEXT X
3060 FOR Y=T TO B
3070 POSITION L(Y),Y
3080 PRINT #6;CHR$(7)
3090 POSITION R(Y),Y
3100 PRINT #6;CHR$(7)
3120 NEXT Y
3130 FOR X=L(Y-1) TO R(Y-1)
3140 POSITION X,Y
3150 PRINT #6;CHR$(7)
3160 NEXT X


3300 FOR Y=T+1 TO P
3310 POSITION X(Y),Y
3320 PRINT #6;CHR$(3+128)
3330 NEXT Y
3340 POSITION 1,20
3350 PRINT #6;"schatz ist bei  x"
3360 POSITION X(P),P
3370 PRINT #6;"X";
3380 POSITION XM,YM
3390 PRINT #6;"$";
3400 RETURN

3500 GOSUB 4500
3510 GET #2,A
3520 IF A=-1 THEN GOTO 3510
3530 POSITION XM,YM
3540 PRINT #6;CHR$(7+128)
```

```
3550 IF A=72 THEN GOSUB 8000:RETURN
3560 IF A=43 THEN XM=XM-1:GOTO 3590
3570 IF A=42 THEN XM=XM+1:GOTO 3590
3580 IF A<>61 THEN GOTO 3500
3590 YM=YM+1
3600 POSITION XM,YM
3610 PRINT #6;"$"
3620 IF MES=0 THEN RETURN
3630 POSITION 0,19
3635 FOR I=1 TO 40
3640 PRINT #6;" ";
3650 NEXT I
3660 MES=0
3670 RETURN

3700 FOR Q=1 TO 10
3710 POSITION V(Q),U(Q)
3720 PRINT #6;CHR$(32+64)
3730 NEXT Q
3740 XS=XS+1
3750 YS=YS+1
3760 RETURN

4000 GOSUB 7000
4010 POSITION XM,YM
4020 PRINT #6;CHR$(32+64)
4030 POSITION XM+1,YM+1
4050 FOR I=60 TO 40 STEP -2
4060 SOUND 0,80-I,10,8
4065 FOR D=1 TO 10:NEXT D
4070 NEXT I
4075 SOUND 0,0,0,0
4080 POSITION 1,19
4090 PRINT #6;"im treibsand"
4095 FOR Q=1 TO 500:NEXT Q
4100 IF RND(0)>0.5 THEN GRAPHICS 1:GOTO 9990
4120 PRINT #6;" -- rausgezogen"
4130 FOR Q=1 TO 500:NEXT Q
4140 RETURN

4500 SOUND 0,80,10,8
4510 FOR D=1 TO 20
4520 NEXT D
4530 SOUND 0,0,0,0
4540 RETURN
```

```
4600 GRAPHICS 1+16
4610 POKE 756,CH/256
4620 SETCOLOR 4,0,0
4630 SETCOLOR 2,0,0
4640 SETCOLOR 0,3,8
4650 SETCOLOR 1,14,8
4660 SETCOLOR 3,12,8
4670 RETURN

5000 L=INT(RND(0)*3)+7
5010 T=INT(RND(0)*2)+2
5020 W=6+INT(RND(0)*3)
5030 B=INT(RND(0)*2)+17
5040 FOR Y=T TO B
5050 L(Y)=L
5060 R(Y)=L+W
5070 L=L-(SGN(10-Y)*INT(RND(0)*2))
5080 W=W+(SGN(10-Y)*INT(RND(0)*2))
5090 IF L(Y)<2 THEN L(Y)=2
5100 IF R(Y)>18 THEN R(Y)=18
5110 NEXT Y

5500 X(T+1)=L(T+1)+INT(RND(0)*3)
5510 K=T+2
5520 FOR P=K TO B-1-INT(RND(0)*3)
5530 X(P)=X(P-1)+INT(RND(0)*3)-1
5540 IF X(P)>=R(P) THEN X(P)=R(P)-1
5550 IF X(P)<=L(P) THEN X(P)=L(P)+1
5560 NEXT P
5570 P=P-1
5580 XT=X(P)
5590 YT=P
5700 FOR Q=1 TO 10
5710 D=INT(RND(0)*(P-T-2))+T+1
5720 U(Q)=D
5730 V(Q)=X(D)+(SGN(RND(0)-0.5)*2)
5740 IF V(Q)<=L(D) THEN V(Q)=V(Q)+3
5750 IF V(Q)>=R(D) THEN V(Q)=V(Q)-3
5760 NEXT Q
5770 RETURN

6000 GOSUB 4600
6020 R=INT(RND(0)*3)
6030 XS=XS+R
6040 YS=YS+R
6050 POSITION 18,18
```

```
6060 PRINT #6;CHR$(32+64)
6070 POSITION XS,YS
6080 PRINT #6;"[";
6090 POSITION XS,YS+1
6100 PRINT #6;"]";
6110 FOR Q=1 TO 200:NEXT Q
6120 IF YS<18 THEN RETURN
6121 GRAPHICS 1
6122 SETCOLOR 2,0,0
6130 POSITION 2,2
6140 PRINT #6;"oweh - die piraten"
6150 POSITION 3,4
6160 PRINT #6;"sind gelandet und"
6165 POSITION 3,6
6166 PRINT #6;"nehmen dich gefangen"
6170 FOR I=1 TO 100:NEXT I
6180 GOTO 9990

7000 GOSUB 4600
7020 FOR X=L(T) TO R(T)
7030 POSITION X,T-1
7040 PRINT #6;CHR$(7+128)
7050 NEXT X
7060 FOR Y=T TO B
7070 POSITION L(Y)-1,Y
7080 PRINT #6;CHR$(7+128)
7090 FOR X=L(Y) TO R(Y)
7100 POSITION X,Y
7110 PRINT #6;CHR$(7+128)
7120 NEXT X
7130 POSITION X,Y
7140 PRINT #6;CHR$(7+128)
7150 NEXT Y
7160 FOR X=L(Y-1) TO R(Y-1)
7170 POSITION X,Y
7180 PRINT #6;CHR$(7+128)
7190 NEXT X
7200 RETURN

8000 GOSUB 3000
8010 GOSUB 3700
8020 FOR Q=1 TO 100+INT(RND(0)*100)
8030 NEXT Q
8040 GOSUB 6000
8050 GOSUB 7000
8060 GOSUB 3600
8070 RETURN
```

```
9000 GRAPHICS 1
9005 FOR C=15 TO 1 STEP -1
9010 SETCOLOR 4,C,8:SETCOLOR 2,C,8
9020 PRINT CHR$(125)
9030 SOUND 0,C+66,10,8
9040 FOR Q=1 TO 20:NEXT Q
9045 NEXT C
9046 SOUND 0,0,0,0
9048 SETCOLOR 0,0,0
9050 POSITION 1,10
9060 PRINT #6;"SCHATZ GEFUNDEN!"

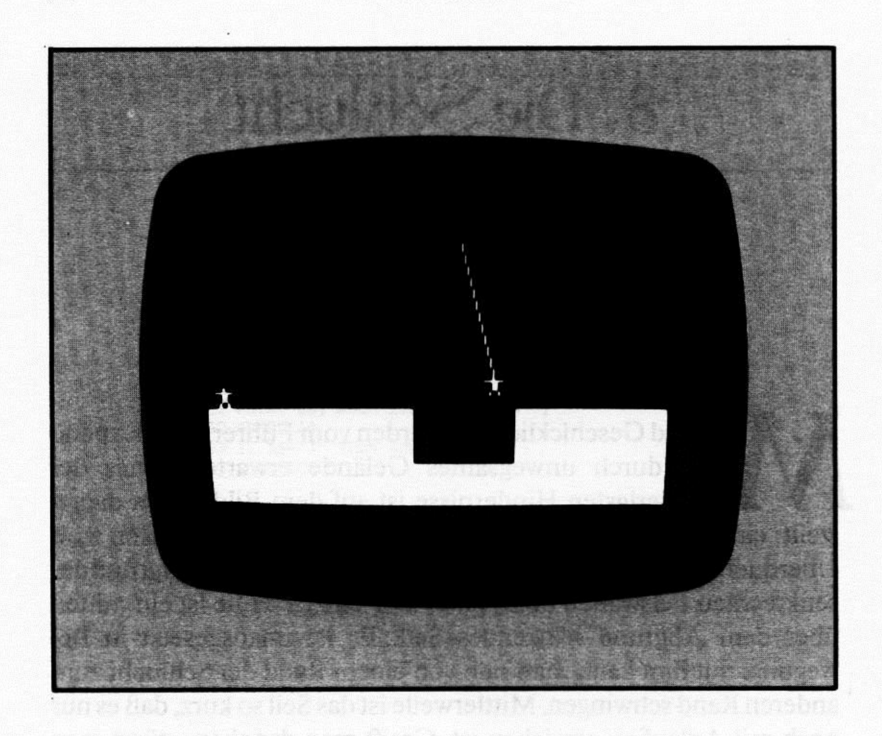
9990 POSITION 1,11
9991 PRINT #6;"NOCH EIN SPIEL J/N ";
9992 INPUT A$
9993 IF A$="J" THEN RUN
9999 GRAPHICS 0
```

8 Die Schlucht

Mut und Geschicklichkeit werden vom Führer einer Expedition durch unwegsames Gelände erwartet. Eines der schwierigsten Hindernisse ist auf dem Bildschirm dargestellt: eine tiefe Schlucht mit einem reißenden Fluß. Brücken zum Überqueren der Schlucht gibt es nicht, Hinabklettern ist aufgrund der senkrechten Felswände unmöglich. Die einzige Hilfe ist ein mitten über dem Abgrund hängendes Seil. Es ist unausgesetzt in Bewegung, mit ihm kann man sich von einem Rand der Schlucht zum anderen Rand schwingen. Mittlerweile ist das Seil so kurz, daß es nur noch mit Anlauf zu erreichen ist. Greift man daneben, stürzt man durch den eigenen Schwung in die Schlucht, was mit dem Leben bezahlt werden muß. Das Spiel ist mit einer eindrucksvollen Geräuschkulisse versehen.

SPIELVERLAUF

Die Expedition besteht aus fünf Teilnehmern. Jeder Teilnehmer wird zum Anlauf am äußeren Bildrand bereitgestellt. Per Tastendruck wird der Anlauf gestartet, der Rest geschieht automatisch. Da man die Sprungkraft jedes Teilnehmers nicht beeinflussen kann, kommt es auf den richtigen Zeitpunkt an, wann der Anlauf gestartet wird.



PROGRAMMAUFBAU

20	Festlegen der Arrays, Hauptprogramm
1500	Pendelvorgang des Seils
2000	Berechnung der Koordinaten für den Pendelvorgang
3000	Schlucht
4000	Ausgabe des sich an dem Seil festhaltenden Expeditionsteilnehmers
5000	Anlauf und Versuch, das Seil zu packen
6000	Bereitstellen des nächsten Sprungkandidaten
7000	Sturz in die Schlucht
7500	Parameter für das Spiel
9000	Ende des Spiels

ERLÄUTERUNGEN

Auch in diesem Programm wird von einigen nachahmenswerten Techniken Gebrauch gemacht.

Die Koordinaten für das schwingende Seil sind für jede Schwingung gleich. Sie werden daher zunächst im Unterprogramm ab Zeile 2000 berechnet und in den Feldern X und Y gespeichert. Durch die so gesparte Rechenzeit für jede Schwingung wird das Programm erheblich schneller. Bei allen periodischen Bewegungen sollte dieses Verfahren angewendet werden.

Für das am Seil hängende Männchen wird die hochauflösende Grafik verwendet, da hier Figuren an jeder beliebigen Stelle des Bildschirms gezeichnet werden können. Durch rasche, nur wenig auseinanderliegende Bildfolgen sind fast ruckfreie Bewegungsabläufe auf dem Bildschirm darstellbar.

```
10 REM Die Schlucht

20 DIM X(16)
30 DIM Y(16)
35 DIM A$(1)
40 GOSUB 3000
50 GOSUB 2000
60 GOSUB 7500
70 GOSUB 6000
75 POKE 764,255
80 GOSUB 1500
85 POKE 764,255
90 IF MEN<=0 THEN MEN=0:POKE 764,255:GOTO 9000
100 GOTO 80

1500 FOR T=1+R TO N-R
1510 COLOR 2:PLOT 85,0
1520 DRAWTO 85+X(T),Y(T)
1525 S=1:COLOR 1:GOSUB 4000
1530 COLOR 0:PLOT 85,0
1540 DRAWTO 85+X(T),Y(T)
1550 IF J=1 THEN S=2:COLOR 0:GOSUB 4000
1560 NEXT T
1565 IF MEN=0 THEN RETURN
1570 IF C=1 THEN GOTO 1680
1600 FOR T=N-R TO 1+R STEP -1
1610 COLOR 2:PLOT 85,0
1620 DRAWTO 85+X(T),Y(T)
1625 S=3:COLOR 1:GOSUB 4000
1630 COLOR 0:PLOT 85,0
1640 DRAWTO 85+X(T),Y(T)
1650 IF J=1 THEN S=4:COLOR 0:GOSUB 4000
1660 NEXT T
1670 RETURN
1680 ACROSS=ACROSS+1
1690 DX=(ACROSS-1)*10+5
1700 DY=52
1710 COLOR 1:GOSUB 4020
1720 C=0
1730 MEN=MEN-1
1735 IF MEN=0 THEN GOTO 6050
1740 GOSUB 6000
1750 RETURN
```

```
2000 N=0
2010 FOR T=-PI/6 TO PI/6 STEP 0.1
2020 N=N+1
2030 X(N)=-INT(50*SIN(T))
2040 Y(N)=INT(50*COS(T))
2050 NEXT T
2060 RETURN


3000 GRAPHICS 7
3010 PRINT "*************************"
3011 PRINT "*                       *"
3012 PRINT "*     Die Schlucht      *"
3013 PRINT "*                       *        ";
3014 REM 8 Leerz.
3020 COLOR 2
3030 PLOT 159,79
3040 DRAWTO 159,60
3050 DRAWTO 105,60
3060 POSITION 105,79
3070 POKE 765,2
3080 XIO 18,#6,0,0,"S:"
3090 PLOT 70,80
3100 DRAWTO 70,60
3110 DRAWTO 0,60
3120 POSITION 0,80
3130 XIO 18,#6,0,0,"S:"
3140 J=0:C=0
3150 PI=4*ATN(1)
3999 RETURN


4000 IF C=0 THEN GOTO 5000
4010 DX=85+X(T):DY=Y(T)
4020 PLOT DX,DY
4030 PLOT DX,DY+1
4040 PLOT DX-3,DY+2
4050 DRAWTO DX+3,DY+2
4060 PLOT DX,DY+3
4070 DRAWTO DX+2,DY+7
4080 PLOT DX+1,DY+3
4090 DRAWTO DX-1,DY+7
4100 RETURN
```

```
5000 A=PEEK(764)
5001 IF A=255 AND J=0 THEN FOR I=1 TO 50:NEXT I:RETURN
5005 POKE 764,255
5006 IF MEN=0 THEN RETURN
5010 J=1
5020 DY=MY
5030 DX=MX
5040 COLOR 0:GOSUB 4020
5050 MX=MX-10
5060 DY=MY
5070 DX=MX
5080 COLOR 1:GOSUB 4020
5090 H=ABS(MX-85-X(T))
5091 IF H<10 AND S=2 THEN C=1:COLOR 0:GOTO 4020
5200 IF MX<105 THEN GOTO 7000
5210 RETURN

6000 MY=52
6010 MX=140
6020 DY=MY
6030 DX=MX
6040 J=0
6045 COLOR 1
6050 GOSUB 4020
6055 PRINT
6060 PRINT "Restl."
6061 PRINT "Gruppe","drueber","abgestuerzt"
6070 PRINT MEN,ACROSS,LOST
6080 RETURN

7000 COLOR 0:GOSUB 4020
7010 MY=MY+2
7020 DY=MY
7030 COLOR 1:GOSUB 4020
7040 SOUND 0,10,MY/2,8
7050 COLOR 0:GOSUB 4020
7060 IF MY<78 THEN GOTO 7010
7070 SOUND 0,80,10,8
7071 FOR I=1 TO 50:NEXT I
7072 SOUND 0,0,0,0
7080 LOST=LOST+1
7090 C=0
7100 J=0
7110 MEN=MEN-1
7120 IF MEN=0 THEN POKE 764,255:GOTO 6060
7130 GOSUB 6000
7140 RETURN
```

```
7500 MEN=5
7510 LOST=0
7520 ACROSS=0
7530 J=0
7540 C=0
7550 R=INT(RND(0)*4)
7560 FOR I=52 TO 59
7570 FOR Q=0 TO 60
7580 COLOR 0:PLOT Q,I
7590 NEXT Q
7595 NEXT I
7596 RETURN


9000 IF LOST=0 THEN GOTO 9009
9001 IF LOST>1 THEN GOTO 9005
9002 PRINT "Es ist einer abgestuerzt":GOTO 9009
9005 PRINT "Es sind ";LOST;" Leute abgestuerzt"
9009 PRINT :PRINT "NOCH EIN SPIEL J/N ";:INPUT A$
9010 IF A$="J" THEN GOTO 60
9020 GRAPHICS 0
```

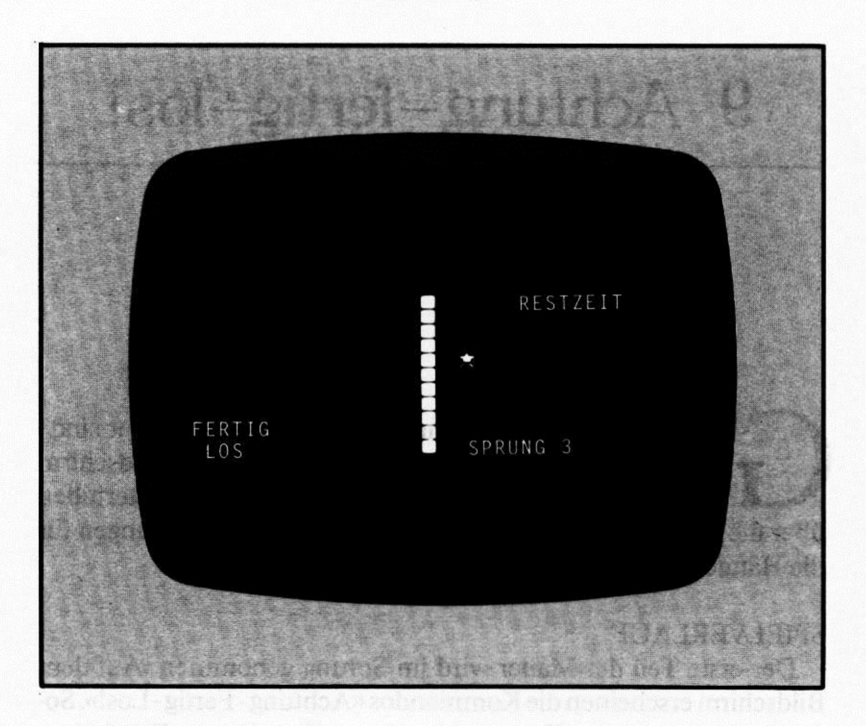
9 Achtung–fertig–los!

Gewandtheit, Reaktionsschnelligkeit und flinke Finger sind bei diesem Spiel gefragt. Das Männchen auf dem Bildschirm soll über eine gewaltige Mauer klettern – ohne Kletterhilfen über die Tastatur ist das nicht zu schaffen. Lockerungsübungen für die Hände werden dringend empfohlen.

SPIELVERLAUF

Der erste Teil der Mauer wird im Sprung genommen. Auf dem Bildschirm erscheinen die Kommandos «Achtung - Fertig - Los!». Sofort nach dem letzten Kommando, aber auf gar keinen Fall früher, wird das Männchen nach einem Tastendruck emporkatapultiert. Die Sprunghöhe hängt von der Zeitspanne zwischen dem Kommando «Los!» und dem Tastendruck ab. Je schneller der Tastendruck erfolgt, um so höher kann das Männchen springen.

Für das Überwinden der Mauer stehen insgesamt 5 Sekunden zur Verfügung. Die nach dem ersten Sprung verbliebene Restzeit wird auf dem Bildschirm angegeben. Zum Weiterklettern sind nun die Auf- und Ab-Pfeiltasten abwechselnd zu drücken. Je schneller die beiden Tasten betätigt werden, um so eher ist das Männchen oben. Wichtig ist, daß die Tasten wirklich abwechselnd gedrückt werden, da nur Tastenpaare gezählt werden. Fünf Paare von Tastendrücken bringen das Männchen um eine Steinreihe höher. Die Zeit läuft unerbittlich weiter! Wird das Ziel in der vorgegebenen Zeit nicht erreicht,



rutscht das Männchen wieder zurück, und ein neuer Versuch kann beginnen.

Da die beiden Pfeiltasten abwechselnd betätigt werden müssen, ist kontinuierliches Festhalten einer solchen Taste sinnlos. Ebenso kann das Männchen nicht schon vor dem letzten Kommando gestartet werden. Mogeln geht also hier nicht!

HINWEISE

Für die Eingabe der Zeilen 8100 bis 8170, die sich nur in einzelnen Zeichen unterscheiden, sollten die Editiermöglichkeiten des Atari ausgenutzt werden.

HINWEISE FÜR ATARI-400-BESITZER

Beim Atari 400 ist es nicht möglich, die Tastendrücke schnell genug zu produzieren. Hier sollte in der Gleichung für die Variable H der Wert 0.2 geändert werden. Dieser Wert bestimmt, wieviel Tastenpaare gezählt werden, bis das Männchen um eine Reihe hochklettern kann. Das Produkt aus der Anzahl der Paare und diesem Wert muß immer 1 ergeben. Also wird für fünf Tastenpaare der Wert 0.2 verwendet, für 2 Tastenpaare wäre die Zahl 0.5 einzusetzen. Durch Ausprobieren kann der geeignetste Wert am einfachsten gefunden werden.

PROGRAMMAUFBAU

20	Hauptteil
1000	Countdown «Achtung - Fertig - Los!»
1500	Test, ob Sprungbefehl zu früh gegeben wurde (Mogelei!)
2000	Mauer
2060	Sprung des Männchens
2500	Zurückrutschen des Männchens bei Zeitüberschreitung
2600	Ziel erreicht
7000	Spielzeit auf Null setzen
7100	Aktuelle Spielzeit
8000	Anfangswerte, Höhe der Mauer
8100	Darstellung des Männchens im RAM-Zeichensatz
8200	Umschalten auf RAM-Zeichensatz und Abschalten des Cursors
9000	Spielstand
9600	Ende des Spiels

ERLÄUTERUNGEN

In diesem Programm wird die Anwendung bewegter, niedrigauflösender Grafik demonstriert. Von Interesse dürfte auch die Verwendung des im Atari eingebauten Taktgebers als Stoppuhr sein. Die Nullstellung dieser Uhr erfolgt in Zeile 7000, das Lesen der Zeit in Zeile 7100. Diese Programmteile können ohne Schwierigkeiten in eigene Programme übertragen werden.

Der Programmteil zum Registrieren eines Tastendrucks ist ebenfalls recht außergewöhnlich. Der Speicherplatz 764 enthält den Code der zuletzt gedrückten Taste. Zum Löschen eines noch dort stehenden Codes wird in den Speicherplatz 764 zunächst über die POKE-Anweisung der Wert 255 (hex FF) eingeschrieben. Jede Änderung dieses Wertes entspricht einem Tastendruck. Es genügt daher, mit Hilfe der PEEK-Anweisung zu testen, ob sich der Wert im Speicherplatz 764 verändert hat. Nach der Registrierung eines Tastendrucks wird zum Löschen wieder der Wert 255 in den Speicherplatz 764 geschrieben. In Zeile 8210 wird der Cursor des Atari abgeschaltet.

```
10 REM Achtung-Fertig-Los
11 REM ------------------

20 GOSUB 8000
30 GOSUB 2000
50 GOTO 9000

1000 POSITION 1,18
1001 PRINT #6;"       ":REM 7 Leerzeichen
1004 PRINT #6;"   ":REM 3 Leerzeichen
1005 POSITION 1,18:PRINT #6;"ACHTUNG"
1010 FOR I=1 TO RND(0)*200+200
1020 NEXT I
1025 POKE 764,255
1030 POSITION 1,18:PRINT #6;" FERTIG"
1040 FOR I=1 TO RND(0)*200+200
1050 NEXT I
1060 IF PEEK(764)<>255 THEN GOTO 1500
1070 GOSUB 7000
1080 PRINT #6;"LOS"
1085 SOUND 0,100,10,10
1090 IF PEEK(764)=255 THEN GOTO 1090
1095 SOUND 0,0,0,0
1100 GOSUB 7100
1110 RETURN

1500 POSITION 1,10:PRINT #6;"GEMOGELT!"
1510 SOUND 0,150,2,8
1515 FOR D=1 TO 1000:NEXT D
1516 SOUND 0,0,0,0
1520 POSITION 1,10:PRINT #6;"         ":REM 9 Leerz.
1530 T=5
1540 POKE 764,255
1550 RETURN

2000 PRINT CHR$(125):JUMP=1
2010 H=10+INT(RND(0)*5)
2020 FOR I=18 TO 19-H STEP -1
2030 POSITION 15,I:PRINT #6;"%"
2040 NEXT I
2050 POSITION 15,18-H:PRINT #6;"%"
```

```
2060 POSITION 18,20:PRINT #6;"SPRUNG ";JUMP
2070 POSITION 18,18:PRINT #6;"$"
2100 GOSUB 1000
2110 FOR I=18 TO 18-H+INT(T*20) STEP -1
2120 POSITION 18,I:PRINT #6;" "
2130 POSITION 18,I-1:PRINT #6;"$"
2140 SOUND 0,80+I,10,10
2145 FOR D=1 TO 10:NEXT D:SOUND 0,0,0,0
2150 NEXT I
2160 J=I:L=INT(I)
2170 GOSUB 7100
2175 IF T>5 THEN GOTO 2500
2180 POSITION 20,9:PRINT #6;"RESTZEIT"
2181 POSITION 22,10
2182 PRINT #6;INT((5-T)*10)/10;"  "
2190 IF PEEK(764)=255 THEN GOTO 2170
2195 POKE 764,255
2200 POSITION 18,L:PRINT #6;" "
2210 J=J-0.2
2215 L=INT(J)
2220 POSITION 18,L:PRINT #6;"$"
2230 IF L>17-H THEN GOTO 2340
2231 POSITION 18,L+1
2233 PRINT #6;" ":GOTO 2600
2340 IF PEEK(764)<>15 THEN GOTO 2340
2350 GOTO 2170


2500 FOR I=L TO 18
2510 POSITION 18,I-1:PRINT #6;" "
2520 POSITION 18,I:PRINT #6;"$"
2530 SOUND 0,80+I,10,10
2535 FOR D=1 TO 5:NEXT D:SOUND 0,0,0,0
2540 NEXT I
2550 JUMP=JUMP+1
2560 POSITION 25,10
2561 PRINT #6;"    ":REM 4 Leerzeichen
2570 IF JUMP<=10 THEN GOTO 2060
2580 RETURN


2600 FOR I=18 TO 10 STEP -1
2610 POSITION I+1,L:PRINT #6;" "
2620 POSITION I,L:PRINT #6;"$"
2630 FOR K=1 TO 10:NEXT K
2640 NEXT I
2650 FOR I=L TO 18
```

```
2660 POSITION 10,I-1:PRINT #6;" "
2670 POSITION 10,I:PRINT #6;"$"
2675 SOUND 0,80+I*2,10,10
2676 FOR D=1 TO 20:NEXT D:SOUND 0,0,0,0
2690 NEXT I
2700 RETURN

7000 POKE 18,0
7010 POKE 19,0
7020 POKE 20,0
7030 RETURN

7100 T=(PEEK(20)+256*PEEK(19))/50
7110 RETURN

8000 GRAPHICS 1+16
8005 POKE 764,255
8010 CH=(PEEK(106)-8)*256
8013 PRINT "Achtung-Fertig-Los":PRINT
8014 PRINT "Ich muss noch einige Vorbereitungen"
8015 PRINT "treffen":PRINT :PRINT
8017 PRINT "Gleich geht´s weiter"
8020 CHORG=(PEEK(756)*256)
8030 FOR I=0 TO 511
8040 POKE CH+I,PEEK(CHORG+I)
8050 NEXT I
8060 POKE CH+(ASC("%")-32)*8+0,0
8070 FOR I=1 TO 7
8080 POKE CH+(ASC("%")-32)*8+I,255
8090 NEXT I
8100 POKE CH+(ASC("$")-32)*8+0,24
8110 POKE CH+(ASC("$")-32)*8+1,24
8120 POKE CH+(ASC("$")-32)*8+2,255
8130 POKE CH+(ASC("$")-32)*8+3,60
8140 POKE CH+(ASC("$")-32)*8+4,60
8150 POKE CH+(ASC("$")-32)*8+5,36
8160 POKE CH+(ASC("$")-32)*8+6,102
8170 POKE CH+(ASC("$")-32)*8+7,195

8200 POKE 756,CH/256
8210 POKE 752,1
8220 T=0
8230 DIM A$(10)
8240 RETURN
```

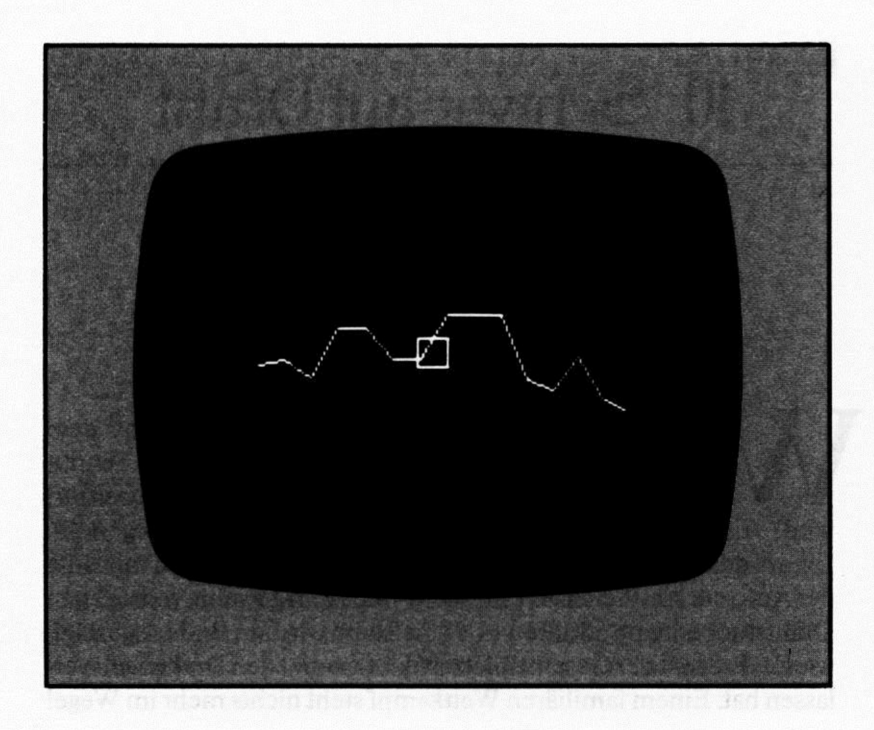
```
9000 IF JUMP<=10 THEN GOTO 9500
9010 POSITION 10,0:PRINT #6;"PECH GEHABT"
9020 GOTO 9600
9500 POSITION 10,0
9501 PRINT #6;"ES WURDEN ";JUMP;" SPRUENGE GEBRAUCHT"
9600 POKE 764,255
9610 POSITION 15,2:PRINT "NOCH EIN SPIEL J/N ";
9620 INPUT A$
9621 IF A$="J" THEN GOTO 30
9640 GRAPHICS 0
```

10 Schwer auf Draht

Wer kennt nicht das beliebte Geschicklichkeitsspiel, bei dem ein Ring über eine Hindernisstrecke so an einer Schnur entlanggeführt werden muß, daß die Schnur nicht berührt wird? Hier wird eine interessante Variante dieses Spiels vorgestellt. Ein auf dem Bildschirm nach rechts wanderndes Quadrat ist mit Hilfe der Auf- und Ab-Pfeiltasten an einem vom Atari gewaltig verbogenen Drahtstück entlangzuführen. Als Spielstand wird am Ende angezeigt, wieviel Prozent der Gesamtspielzeit das Quadrat den Draht nicht verlassen hat. Einem familiären Wettkampf steht nichts mehr im Wege!

SPIELVERLAUF

Zu Beginn des Spiels ist der Schwierigkeitsgrad zu wählen, von dem die Größe des Quadrats abhängt. Das Quadrat bleibt einen Augenblick am Anfang der Linie stehen, startet dann automatisch und bewegt sich mit konstanter Geschwindigkeit nach rechts. Durch Betätigen der Auf- und Ab-Pfeiltasten kann das Quadrat nach oben und unten gelenkt werden (Autorepeatfunktion der Tasten ausnutzen).



PROGRAMMAUFBAU

15	Hauptprogramm
110	Ende des Spiels
1000	Anfangswerte und Titelbild
2000	Zackenlinie
5000	Steuerung des Quadrats, Reparatur der Linienteile, die von dem darüber hinweg wandernden Quadrat gelöscht worden sind
5570	Test, ob das Quadrat auf der Linie läuft
6000	Tonerzeugung

ERLÄUTERUNGEN

Um eine kontinuierliche Bewegung des Quadrats zu ermöglichen, wird die hochauflösende Grafik verwendet. Im Gegensatz zur Blockgrafik können hier Figuren punktweise verschoben werden, wodurch abgehackte Bewegungen vermieden werden. Das Unterprogramm ab Zeile 2000 konstruiert den verbogenen Draht, das Quadrat wird mit Hilfe der DRAWTO-Anweisungen ab Zeile 5000 gezeichnet. Die LOCATE-Funktion in Zeile 5570 testet, ob die Linie während des Spiels berührt wurde.

```
10 REM Schwer auf Draht

15 DIM A$(10),C(15)
20 GOSUB 1000
30 GOSUB 2000
40 FOR Z=1 TO 100:NEXT Z
60 GOSUB 6000
70 Y=35:X=10
80 B=Y:R=12-DF
90 R2=INT(R/2):V=2
100 GOSUB 5000

110 PRINT "Der Draht wurde ";
111 PRINT INT(HIT/(HIT+MISS)*10000)/100;
112 PRINT "% der Spielzeit"
113 PRINT "nicht verlassen."
120 PRINT "Noch ein Spiel J/N ";:INPUT A$
130 IF A$="J" THEN RUN
140 IF A$<>"N" THEN GOTO 120
160 PRINT CHR$(125)
170 STOP


1000 X=10
1010 Y=40
1020 D=30
1030 P=0
1070 PRINT CHR$(125)
1080 POSITION 10,2:PRINT "Spielregel:"
1090 PRINT :PRINT "Auf dem Bildschirm wird ein gewaltig"
1095 PRINT "verbogener Draht gezeichnet."
1096 PRINT
1100 PRINT "Das Quadrat muss so gesteuert werden,"
1105 PRINT "dass es immer auf dem Draht bleibt."
1109 PRINT
1110 PRINT "AUF- und AB-Pfeil benutzen!"
1120 POSITION 0,16
1121 PRINT "Die Gesamtbewertung erfolgt am Schluss"
1150 POSITION 5,21:PRINT "Schwierigkeitsgrad"
1151 POSITION 5,22:PRINT "<1> leicht ... <5> schwer ";
1155 INPUT DF
1170 IF DF<1 OR DF>5 THEN GOTO 1150
1180 PRINT CHR$(125)
1190 GRAPHICS 7+16:COLOR 1
1210 POKE 752,1
1220 COLOR 1
1999 RETURN
```

```
2000 PLOT X,Y
2005 C(0)=Y
2010 FOR I=1 TO 14
2020 R=D-INT(RND(0)*2*D)
2030 Y=Y+R:X=X+10
2040 IF Y>60 THEN Y=Y-R
2050 IF Y<20 THEN Y=Y-R
2060 DRAWTO X,Y
2065 C(I)=Y
2070 NEXT I
2080 HIT=0
2090 MISS=0
2100 RETURN


5000 PLOT X-R2,Y-R2
5010 DRAWTO X-R2,R+Y-R2
5020 DRAWTO X-R2+R,R+Y-R2
5030 DRAWTO X-R2+R,Y-R2
5040 DRAWTO X-R2,Y-R2
5050 PLOT 10,C(0)
5060 FOR I=1 TO 14
5070 DRAWTO 10+I*10,C(I)
5080 NEXT I
5100 M=PEEK(764)
5105 POKE 764,255
5110 IF M=14 AND Y-R2>2*V THEN B=B-2*V
5120 IF M=15 AND Y+R<80-2*V THEN B=B+2*V
5500 COLOR 0
5510 PLOT X-R2,Y-R2
5520 DRAWTO X-R2,R+Y-R2
5530 DRAWTO X-R2+R,R+Y-R2
5540 DRAWTO X-R2+R,Y-R2
5550 DRAWTO X-R2,Y-R2
5560 X=X+V
5561 COLOR 1
5565 IF X>145 THEN RETURN

5570 Y=B
5575 H=0
5580 FOR I=-R2 TO R2
5585 LOCATE X,Y+I,Q
5590 IF Q<>0 THEN HIT=HIT+1:FOR Z=1 TO 20:NEXT Z:I=R-1:H=1
5600 NEXT I
5605 IF H=1 THEN GOTO 5000
5610 MISS=MISS+1
5700 GOSUB 6000
5710 GOTO 5000
```

```
6000 SOUND 0,80,10,8
6010 FOR Q=1 TO 10
6020 NEXT Q
6030 SOUND 0,0,0,0
6040 RETURN
```



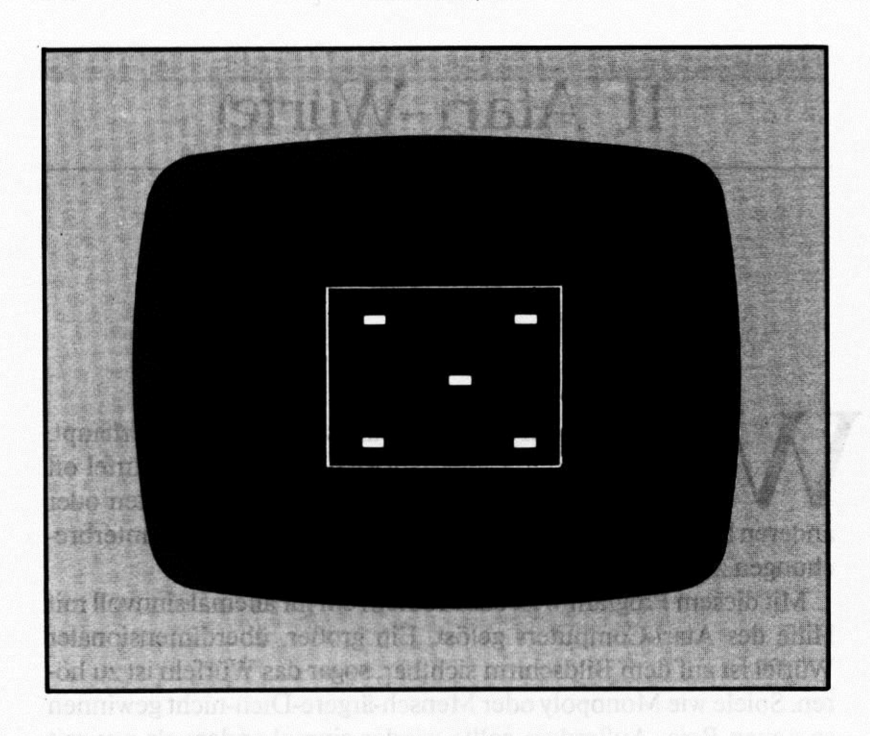
11 Atari-Würfel

Würfelspiele gehören zu den ältesten Spielen überhaupt. Eines der Hauptprobleme ist jedoch, daß der Würfel oft vom Tisch rollt und erst mühsam unter Schränken oder anderen Möbeln gesucht werden muß, was ärgerliche Spielunterbrechungen zur Folge hat.

Mit diesem Programm ist das Problem ein für allemal sinnvoll mit Hilfe des Atari-Computers gelöst. Ein großer, überdimensionaler Würfel ist auf dem Bildschirm sichtbar, sogar das Würfeln ist zu hören. Spiele wie Monopoly oder Mensch-ärgere-Dich-nicht gewinnen so neuen Reiz. Außerdem sollte wieder einmal anders als nur mit dem Computer gespielt werden.

ANWENDUNG DES PROGRAMMS

Nach dem üblichen RUN wird das Würfeln (Promptzeichen abwarten) durch einen Tastendruck gestartet. Während der Würfel rollt, erscheinen auf dem Bildschirm die «oben liegenden» Punktzahlen. Am Anfang kommen diese Zahlen sehr rasch, dann rollt der Würfel langsam aus. Wie üblich, hilft die RND-Funktion bei diesen Vorgängen. Das Programm kann über die BREAK-Taste verlassen werden.



PROGRAMMAUFBAU

10 Hauptprogramm
1000 Ausgabe bzw. Löschen der Würfelpunkte auf dem Bildschirm
2000 Würfelfläche (gelbes Quadrat)
5000 Farben; Bereitstellen der Tastatur als INPUT-File
6000 Würfelgeräusche
7000 Kopiert Zeichensatz in den RAM-Bereich
7060 Darstellung der Würfelpunkte im RAM-Zeichensatz
7090 Anfangswerte der Variablen
7200 Umschalten auf den RAM-Zeichensatz

ERLÄUTERUNGEN

Ein Würfelprogramm ist die ideale Anwendung von Zufallszahlen. In diesem Programm wird hiervon gleich zweimal Gebrauch gemacht. Zunächst wird über die RND-Funktion in Zeile 140 bestimmt, welche Zahl während des Würfelns als nächste auf dem Bildschirm erscheinen soll. Diese Zufallszahl muß entsprechend den möglichen Würfelpunkten zwischen 1 und 6 liegen. Ihr Wert wird dem Ausgabeprogramm (ab Zeile 1000) übergeben, das je nach Wert der Zahl zu den Zeilen 1100 (1 Würfelpunkt), 1200 (2 Punkte) usw. verzweigt. Auf dem Bildschirm erscheint die richtige Augenzahl des Würfels.

Die zweite Anwendung der RND-Funktion (Zeilen 90 bis 110) bestimmt das Ausrollen des Würfels. Die Variable T nimmt Werte zwischen 5 und 15 an und legt so die Zahl der Drehungen des Würfels fest, bevor der Ausrollvorgang beginnt.

EIGENE ERWEITERUNGEN

Das Programm kann ohne Schwierigkeiten in andere Programme eingebaut werden. So könnten die Wetten im Spiel «Pferderennen» auf gewürfelten Nummern gesetzt werden.

```
5 REM Atari-Wuerfel

10 GOSUB 7000
12 PRINT #6;CHR$(125)
20 GOSUB 5000
30 GOSUB 2000
40 POSITION 0,22
50 PRINT #6;"IRGENDEINE TASTE"
51 PRINT #6;"DRUECKEN"
60 GET #2,B
90 T=RND(0)*5+5
100 FOR I=1 TO T
110 FOR Q=1 TO 20*I:NEXT Q
120 D$=B$
130 GOSUB 1000
140 R=INT(RND(0)*6)+1
150 D$=A$
160 GOSUB 1000
170 NEXT I
180 GOTO 60


1000 GOSUB 6000
1010 GOTO 1000+R*100
1100 POSITION 8,11
1110 PRINT #6;D$
1120 RETURN
1200 POSITION 5,5
1210 PRINT #6;D$
1220 POSITION 12,17
1230 PRINT #6;D$
1240 RETURN
1300 GOSUB 1100
1310 GOSUB 1200
1400 POSITION 5,17
1410 PRINT #6;D$
1420 POSITION 12,5
1430 PRINT #6;D$
1440 GOTO 1200
1500 GOSUB 1400
1510 GOTO 1100
1600 POSITION 5,11
1610 PRINT #6;D$
1620 POSITION 12,11
1630 PRINT #6;D$
1640 GOTO 1400
```

```
2000 FOR I=1 TO 16
2010 POSITION 4,I+2
2020 PRINT #6;B$
2030 NEXT I
2040 R=1
2050 D$=A$
2060 GOTO 1000


5000 REM Farben
5010 SETCOLOR 0,0,0
5020 SETCOLOR 1,12,12
5030 SETCOLOR 2,3,6
5040 SETCOLOR 4,8,10
5050 OPEN #2,4,0,"K"
5060 RETURN

6000 SOUND 0,100,12,8
6010 SOUND 0,0,0,0
6020 RETURN

7000 GRAPHICS 1+16
7004 POSITION 4,8
7005 PRINT #6;"ATARI-WUERFEL"
7006 POSITION 0,18
7007 PRINT #6;"GLEICH GEHT´S WEITER"
7010 CH=(PEEK(106)-8)*256
7020 CHORG=(PEEK(756)*256)
7030 FOR I=0 TO 511
7040 POKE CH+I,PEEK(CHORG+I)
7050 NEXT I

7060 FOR I=0 TO 7
7070 POKE CH+(ASC("$")-32)*8+I,255
7080 NEXT I

7090 DIM A$(1),B$(10),D$(1)
7100 A$=CHR$(164)
7110 FOR I=1 TO 10
7120 B$(I)=CHR$(4)
7130 NEXT I

7200 POKE 756,CH/256
7999 RETURN
```

12 Invasion der Außerirdischen

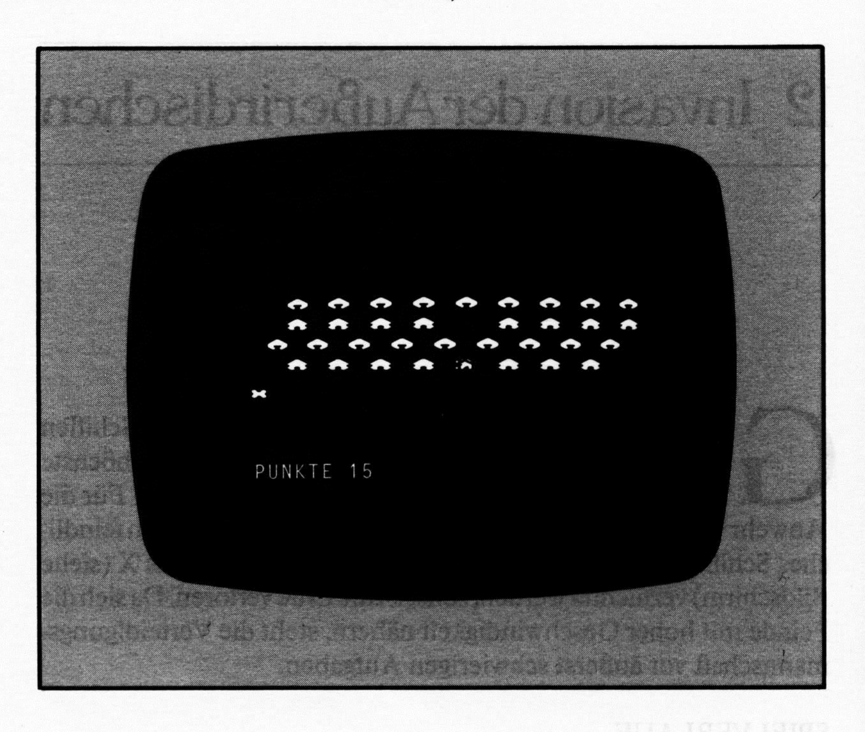
Großangriff auf die Erde! Pulks von roten und gelben Schiffen außerirdischer Eindringlinge rasen auf die Erde zu – höchste Alarmbereitschaft für die Verteidigungsstreitkräfte! Für die Abwehr werden Raketen eingesetzt, jeder Treffer zerstört ein feindliches Schiff. Alle Schiffe müssen vor Erreichen des Punkts X (siehe Bildschirm) vernichtet werden, sonst ist die Erde verloren. Da sich die Feinde mit hoher Geschwindigkeit nähern, steht die Verteidigungsmannschaft vor äußerst schwierigen Aufgaben.

SPIELVERLAUF

Die Raketenabschußbasis der Erde wird über die Rechts- und Linkspfeiltasten des Atari bewegt, der Raketenabschuß erfolgt über die Aufwärts-Pfeiltaste. Ein Punktezähler registriert jeden Abschuß, wobei die erreichte Punktezahl von der Entfernung des getroffenen Schiffes abhängt. Das Spiel ist zu Ende, wenn entweder alle gegnerischen Schiffe zerstört worden sind oder wenn mindestens eines der feindlichen Schiffe den Punkt X passieren konnte.

PROGRAMMAUFBAU

20 Grafikmodus
140 Stringvariablen
220 Anfangswerte der Strings
280 Hauptprogramm



580 Ende des Spiels
740 Bewegung der Raketenabschußbasis, Abfeuern einer Rakete
890 Berechnungen für einen Raketenabschuß
1250 Test auf Treffer
1360 Bewegung der gegnerischen Schiffe nach rechts
1380 Bewegung der gegnerischen Schiffe nach links
2000 Darstellung eines gegnerischen Schiffes (Typ I) im RAM-Zeichensatz
2100 Darstellung der Raketenabschußbasis im RAM-Zeichensatz
2200 Darstellung eines gegnerischen Schiffes (Typ II) im RAM-Zeichensatz
2300 Darstellung der Explosion eines gegnerischen Schiffes im RAM-Zeichensatz

2400 Umschalten des Atari auf den RAM-Zeichensatz
2500 Tonerzeugung
2600 Darstellung der gegnerischen Schiffssteuerung

ERLÄUTERUNGEN

Die gegnerischen Schiffe werden in Stringvariablen gespeichert. Zur Darstellung der beiden Schiffstypen sind das $-Zeichen und das &-Zeichen im Zeichensatz neu definiert worden. Die Frontlinie (Zeile 220) wird durch acht Schiffe des Typs $ dargestellt, die zweite Reihe durch acht Schiffe des &-Typs (Zeile 230). Um die zweite Reihe gegen die erste versetzt darzustellen, sind Leerzeichen eingefügt. Die dritte und vierte Reihe sind Kopien der anderen beiden Reihen (Zeilen 240 und 250). In Zeile 260 werden für jede Angriffsreihe entsprechend lange Strings aus Leerzeichen gebildet. Mit diesen Leerzeichenstrings wird in den Zeilen 520 bis 570 getestet, ob feindliche Schiffe den Punkt X überwinden konnten.

Durch die versetzten Bewegungen der gegnerischen Angriffsreihen erhält das Spiel eine zusätzliche Note. In Zeile 1260 wird geprüft, ob ein feindliches Schiff von einer Rakete getroffen worden ist. In diesem Fall wird der String, in dem der Treffer registriert worden ist, an der entsprechenden Stelle geteilt, das zerstörte Schiff durch ein Leerzeichen ersetzt und die zwei Hälften des Strings wieder zusammengesetzt. Dieses Programm demonstriert anschaulich die im Atari-BASIC möglichen String-Manipulationen.

EIGENE ERWEITERUNGEN

Noch spannender wird das Spiel, wenn auch die Gegner schießen können. Für die Raketenstation der Erde müssen dann Abwehrmaßnahmen (Ausweichmanöver, Schutzschild) vorgesehen werden.

```
10 REM Invasion der Ausserirdischen

20 GRAPHICS 1+16
25 POSITION 3,3:PRINT #6;"INVASION DER"
26 POSITION 2,6:PRINT #6;"AUSSERIRDISCHEN"
27 POSITION 0,12
28 PRINT #6;"GLEICH GEHT´S WEITER"
30 OPEN #2,4,0,"K"
40 CH=(PEEK(106)-8)*256
50 CHORG=PEEK(756)*256
60 FOR I=0 TO 511
70 POKE CH+I,PEEK(CHORG+I)
80 NEXT I
100 SETCOLOR 0,3,6
110 SETCOLOR 1,0,14
120 SETCOLOR 2,13,8
130 GOSUB 2000:PRINT #6;CHR$(125)

140 DIM A$(20):DIM B$(20)
144 DIM C$(20):DIM D$(20)
146 DIM E$(20):DIM Q$(20)
148 DIM T$(20)
150 Y=1
160 XL=10
170 YM=0
180 T=0
190 S=0
200 J=0
210 K=0

220 FOR I=0 TO 8
225 A$(1+I*2)=CHR$(164)
226 A$(2+I*2)=" "
228 NEXT I
230 B$=" & & & & & & & & &"
240 C$=A$
250 D$=B$
260 E$="                  ":REM 18 Leerzeichen

280 POSITION 0,14:PRINT #6;"X";
290 J= NOT J
300 IF K=1 THEN GOTO 600
310 IF T>30+INT(RND(0)*15) THEN POSITION 0,Y:PRINT #6;E$:Y=Y+2:T=0:GOSUB 2500
315 Q$=A$
320 IF J THEN GOSUB 1360:A$=Q$:GOTO 340
```

```
330 GOSUB 1380:A$=Q$
340 POSITION 1,Y:PRINT #6;A$
350 SOUND 1,10,50+Y*8,10
360 GOSUB 740
365 Q$=B$
370 IF J THEN GOSUB 1360:B$=Q$:GOTO 390
380 GOSUB 1380:B$=Q$
390 POSITION 1,Y+2:PRINT #6;B$
400 SOUND 1,10,129-Y*8,10
410 GOSUB 740
420 Q$=C$:GOSUB 1360:C$=Q$
440 POSITION 1,Y+4:PRINT #6;C$
450 SOUND 1,5,121-Y*8,5:SOUND 0,0,0,0
460 GOSUB 740
470 Q$=D$:GOSUB 1380:D$=Q$
490 POSITION 1,Y+6:PRINT #6;D$
500 SOUND 1,10,129-Y*8,10
510 GOSUB 740
520 IF Y>8 AND D$<>E$ THEN GOTO 580
530 IF Y>10 AND C$<>E$ THEN GOTO 580
540 IF Y>12 AND B$<>E$ THEN GOTO 580
550 IF Y>14 AND A$<>E$ THEN GOTO 580
560 T=T+1
565 GOSUB 2600
570 GOTO 280

580 POSITION 3,18
581 PRINT #6;"DIE WELT IST"
582 POSITION 3,19
583 PRINT #6;"VERLOREN"
590 GOTO 610
600 POSITION 1,18
601 PRINT #6;"GUT GEMACHT!"
602 POSITION 1,19
603 PRINT #6;"WELT GERETTET"
610 SOUND 0,1,80,10
620 FOR D=1 TO 1000:NEXT D
650 IF Q=0 THEN SOUND 0,10,4,10
655 SOUND 0,0,0,0
660 PRINT "NOCH EIN SPIEL J/N ";:INPUT A$
680 IF A$="J" THEN RUN
730 END
```

```
740 A=PEEK(764)
750 POKE 764,255
770 T=T+1
790 POSITION XL,20:PRINT #6;CHR$(5)
800 IF A=255 THEN RETURN
810 POSITION XL,20:PRINT #6;" "
820 IF A=6 AND XL>1 THEN XL=XL-1
830 IF A=7 AND XL<17 THEN XL=XL+1
850 POSITION XL,20:PRINT #6;CHR$(5)
860 IF A=14 THEN GOSUB 890
880 RETURN

890 REM Schiessen
900 FOR M=18 TO Y+6 STEP -1
910 POSITION XL,M:PRINT #6;":";
920 POSITION XL,M+1:PRINT #6;" "
930 NEXT M
940 POSITION XL,M+1:PRINT #6;" "
950 F=0
960 Q$=D$
970 R=6
980 GOSUB 1250
990 D$=Q$
1000 IF F=1 THEN GOTO 1220
1020 POSITION XL,Y+5:PRINT #6;".";
1021 POSITION XL,Y+5:PRINT #6;" ";
1025 POSITION XL,Y+4:PRINT #6;".";
1026 POSITION XL,Y+4:PRINT #6;" ";
1030 Q$=C$
1040 R=4
1050 GOSUB 1250
1060 C$=Q$
1070 IF F=1 THEN GOTO 1220
1080 POSITION XL,Y+3:PRINT #6;".";
1081 POSITION XL,Y+3:PRINT #6;" ";
1090 POSITION XL,Y+2:PRINT #6;".";
1091 POSITION XL,Y+2:PRINT #6;" "
1100 Q$=B$
1110 R=2
1130 GOSUB 1250
1140 B$=Q$
1150 IF F=1 THEN GOTO 1220
1160 POSITION XL,Y+1:PRINT #6;".";
1161 POSITION XL,Y+1:PRINT #6;" ";
1165 POSITION XL,Y:PRINT #6;".";
1166 POSITION XL,Y:PRINT #6;" ";
1170 Q$=A$
```

```
1180 R=0
1200 GOSUB 1250
1210 A$=Q$
1220 IF A$=E$ AND B$=E$ AND C$=E$ AND D$=E$ THEN K=1
1230 IF Q$=E$ THEN POSITION 1,Y:PRINT #6;E$:Y=Y+2
1240 RETURN

1250 REM getroffen!
1260 IF Q$(XL,XL)=" " THEN RETURN
1270 Q$(XL,XL)=" "
1280 F=1
1290 S=S+10-Y
1310 POSITION XL,Y+R:PRINT #6;"@"
1320 SOUND 1,15,4,3:SOUND 0,0,0,0
1330 POSITION 2,21:PRINT #6;"PUNKTE ";S;" "
1340 T=T-RND(0)*3
1350 RETURN

1360 QL=LEN(Q$):Q$(QL+1)=Q$(1,1):Q$=Q$(2,QL+1)
1370 RETURN

1380 T$=Q$(LEN(Q$))
1390 T$(2)=Q$(1,LEN(Q$)-1)
1395 Q$=T$
1398 RETURN

2000 POKE CH+(ASC("$")-32)*8+0,24
2010 POKE CH+(ASC("$")-32)*8+1,60
2020 POKE CH+(ASC("$")-32)*8+2,126
2030 POKE CH+(ASC("$")-32)*8+3,255
2040 POKE CH+(ASC("$")-32)*8+4,195
2050 POKE CH+(ASC("$")-32)*8+5,195
2060 POKE CH+(ASC("$")-32)*8+6,102
2070 POKE CH+(ASC("$")-32)*8+7,36

2100 POKE CH+(ASC("%")-32)*8+0,24
2110 POKE CH+(ASC("%")-32)*8+1,24
2120 POKE CH+(ASC("%")-32)*8+2,24
2130 POKE CH+(ASC("%")-32)*8+3,60
2140 POKE CH+(ASC("%")-32)*8+4,126
2150 POKE CH+(ASC("%")-32)*8+5,126
2160 POKE CH+(ASC("%")-32)*8+6,255
2170 POKE CH+(ASC("%")-32)*8+7,255
```

```
2200 POKE CH+(ASC("&")-32)*8+0,24
2210 POKE CH+(ASC("&")-32)*8+1,60
2220 POKE CH+(ASC("&")-32)*8+2,126
2230 POKE CH+(ASC("&")-32)*8+3,255
2240 POKE CH+(ASC("&")-32)*8+4,60
2250 POKE CH+(ASC("&")-32)*8+5,102
2260 POKE CH+(ASC("&")-32)*8+6,195
2270 POKE CH+(ASC("&")-32)*8+7,102

2300 POKE CH+(ASC("@")-32)*8+0,40
2310 POKE CH+(ASC("@")-32)*8+1,132
2320 POKE CH+(ASC("@")-32)*8+2,145
2330 POKE CH+(ASC("@")-32)*8+3,40
2340 POKE CH+(ASC("@")-32)*8+4,28
2350 POKE CH+(ASC("@")-32)*8+5,52
2360 POKE CH+(ASC("@")-32)*8+6,164
2370 POKE CH+(ASC("@")-32)*8+7,164

2400 POKE 756,CH/256
2410 POKE 752,1
2420 FLAG=0
2499 RETURN

2500 SOUND 1,4,50,8:SOUND 0,0,0,0
2510 RETURN

2600 IF FLAG=1 THEN GOTO 2650
2610 POKE CH+(ASC("$")-32)*8+7,129
2620 POKE CH+(ASC("&")-32)*8+7,195
2630 FLAG=1
2640 RETURN
2650 POKE CH+(ASC("$")-32)*8+7,36
2660 POKE CH+(ASC("&")-32)*8+7,102
2670 FLAG=0
2680 RETURN
```

13 Atari–Mühle

Gegenspieler bei diesem etwas abgewandelten Mühlespiel ist der Computer mit seinen wohldurchdachten Zügen. Da aber keine Analyse für die weiteren Spielzüge durchgeführt wird, erweist sich der Atari als fairer Gegenspieler. Die Chancen, gegen den Computer zu gewinnen, sind erstaunlich gut! Viel Spaß bei diesem lehrreichen Strategiespiel!

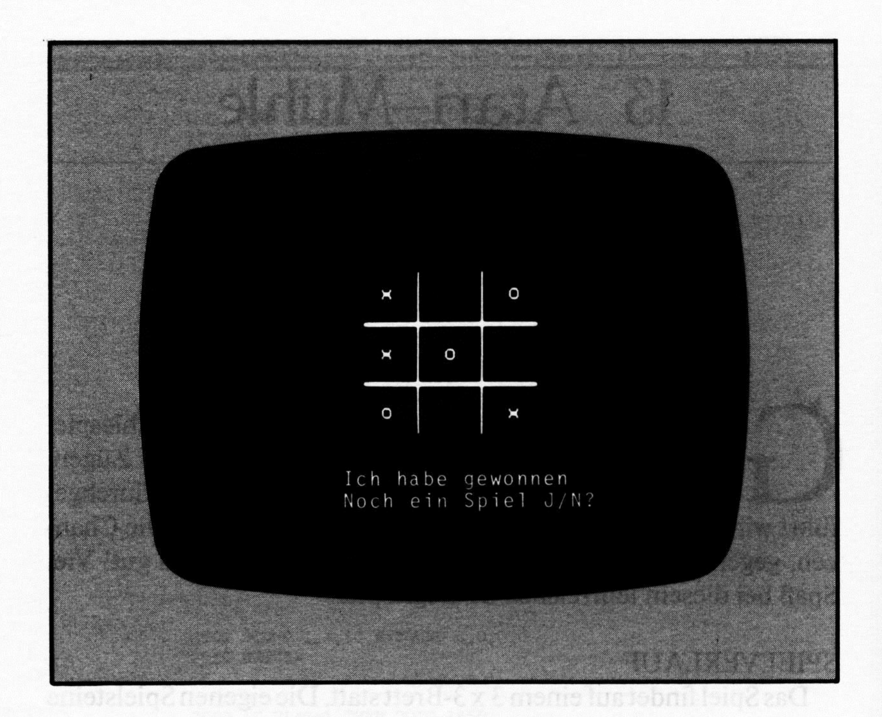
SPIELVERLAUF

Das Spiel findet auf einem 3 x 3-Brett statt. Die eigenen Spielsteine sind mit X (Zeile 7020), die des Atari mit O (Zeile 7030) bezeichnet. Man hat stets den ersten Zug.

Die Angabe des Feldes, auf dem der Stein plaziert werden soll, erfolgt durch Eingabe der Spalten- und Zeilennummer, wobei zwischen den beiden Zahlen weder ein Zwischenraum noch ein anderes Zeichen stehen darf.

ANORDNUNG DER FELDER:

		Spalte 1	2	3
	1	11	21	31
Zeile	2	12	22	32
	3	13	23	33



Um einen Spielstein in die linke obere Ecke zu setzen, ist also 11 einzugeben. Falscheingaben werden nicht angenommen, bereits besetzte Felder können nicht noch einmal belegt werden. Nach dem eigenen Zug setzt der Computer seinen Spielstein. Wenn eine Mühle gebildet oder das gesamte Spielfeld mit Steinen besetzt worden ist, wird das Spiel beendet und auf dem Bildschirm das Ergebnis angezeigt.

HINWEISE

Zur Darstellung des Spielfelds werden einige der Grafikzeichen des Atari verwendet. Da diese Zeichen nicht im Programmlisting gedruckt werden können, sind sie in die spitzen Klammern < und > gesetzt (Zeilen 8190 und 8400). So ist z.B. in der Zeile 8190 die Grafik-

Buchstabenfolge RRRSRRRSRRR einzugeben, im Listing steht also <RRRSRRRSRRR>. Beim Erreichen der Klammer < ist der Atari zunächst in den Grafik-Modus zu schalten (Taste CAPS und CTRL gemeinsam drücken), dann wird die Buchstabenfolge eingetippt. Die Klammer > bedeutet Abschalten des Grafik-Modus (CAPS und SHIFT gemeinsam drücken). Auf dem Bildschirm erscheint anstelle der Grafikbuchstaben das Gitter des Spielfelds.

Den senkrechten Strich der Zeilen 8200 bis 8390 erreicht man durch gleichzeitiges Drücken der SHIFT- und der =-Taste (Gleichheitszeichen).

PROGRAMMAUFBAU

20	Hauptprogramm
4000	Berechnung des Computerzuges
5000	Jeder mögliche Zug wird berücksichtigt
6000	eigener Zug
7000	Ausgabe der Spielfelder auf dem Bildschirm
8000	Initialisierung der Felder und Variablen
8140	Ausgabe des Spielbretts
8500	Tonerzeugung
9000	Spielende

ERLÄUTERUNGEN

Der Computer berechnet nur seinen nächsten Zug. Er berücksichtigt nicht, welche Möglichkeiten sich in den folgenden Zügen noch ergeben könnten. Die Chancen, gegen den Computer zu gewinnen, sind daher recht gut. Trotz dieser bewußten Einschränkung für die Computerspielzüge basiert das Programm auf den Prinzipien der künstlichen Intelligenz.

```
10 REM Atari-Muehle

20 GOSUB 8000
30 GOSUB 6000
40 GOSUB 7000
50 GOSUB 5000
60 IF FIN=1 THEN GOTO 9500
70 IF FIN=2 THEN GOSUB 7000:GOTO 9000
75 IF DR=1 THEN GOTO 9100
80 GOSUB 7000
90 GOTO 30
99 STOP


4000 FOR Q=1 TO 4
4010 X(Q)=0:Y(Q)=0
4015 NEXT Q
4020 FOR L=1 TO 3
4030 S=0
4040 T=0
4050 FOR K=1 TO 3
4060 IF A(L,K)=1 THEN S=S+1
4070 IF B(L,K)=1 THEN T=T+1
4080 NEXT K
4090 IF S=0 THEN Y(T+1)=Y(T+1)+1
4100 IF T=0 THEN X(S+1)=X(S+1)+1
4105 NEXT L
4110 FOR L=1 TO 3
4120 T=0
4125 S=0
4130 FOR K=1 TO 3
4140 IF A(K,L)=1 THEN S=S+1
4150 IF B(K,L)=1 THEN T=T+1
4160 NEXT K
4170 IF S=0 THEN Y(T+1)=Y(T+1)+1
4180 IF T=0 THEN X(S+1)=X(S+1)+1
4185 NEXT L
4190 GOSUB 4300
4200 GOSUB 4400
4210 IF X(4)=1 THEN FIN=1:RETURN
4215 IF Y(4)=1 THEN FIN=2
4220 E=128*Y(4)-63*X(3)+31*Y(3)-15*X(2)+7*Y(2)
4230 RETURN
4300 T=0
4310 S=0
4320 FOR K=1 TO 3
4330 T=T+A(K,K)
```

```
4340 S=S+B(K,K)
4350 NEXT K
4360 IF S=0 THEN X(T+1)=X(T+1)+1
4370 IF T=0 THEN Y(S+1)=Y(S+1)+1
4380 RETURN

4400 T=0
4410 S=0
4420 FOR K=1 TO 3
4430 T=T+A(4-K,K)
4440 S=S+B(4-K,K)
4450 NEXT K
4460 IF S=0 THEN X(T+1)=X(T+1)+1
4470 IF T=0 THEN Y(S+1)=Y(S+1)+1
4480 RETURN

5000 M=-256:DR=1
5005 FOR J=1 TO 3
5010 FOR I=1 TO 3
5015 IF A(I,J)=1 OR B(I,J)=1 THEN GOTO 5040
5016 DR=0:B(I,J)=1
5020 GOSUB 4000
5025 IF FIN=1 THEN RETURN
5030 IF E>M THEN M=E:A=I:B=J
5035 B(I,J)=0
5040 NEXT I
5050 NEXT J
5060 B(A,B)=1
5070 RETURN

6000 POSITION 8,19:PRINT "Dein Zug (Spalte/Reihe) "
6001 INPUT A$
6005 IF LEN(A$)<>2 THEN GOSUB 8500:GOTO 6000
6010 J=VAL(A$(1,1)):I=VAL(A$(2,2))
6020 IF I<1 OR I>3 THEN GOSUB 8500:GOTO 6000
6030 IF I<1 OR J>3 THEN GOSUB 8500:GOTO 6000
6040 IF A(I,J)=1 THEN GOTO 6100
6050 IF B(I,J)=1 THEN GOTO 6100
6060 A(I,J)=1
6070 POSITION 8,20
6071 PRINT "                              "
6073 REM *     30 Leerzeichen
6080 POSITION 8,19
6081 PRINT "                               "
6083 REM *     31 Leerzeichen
```

```
6090 RETURN
6100 POSITION 8,20
6105 PRINT "Dieser Platz ist schon belegt"
6110 GOSUB 8500
6120 GOTO 6000

7000 FOR J=1 TO 3
7010 FOR I=1 TO 3
7020 IF A(I,J)=1 THEN POSITION J*4+8,I*4+3:PRINT "X";
7030 IF B(I,J)=1 THEN POSITION J*4+8,I*4+3:PRINT "O";
7040 IF A(I,J)+B(I,J)<>0 THEN GOTO 7050
7045 POSITION J*4+8,I*4+3:PRINT " ";
7050 NEXT I
7060 PRINT
7070 NEXT J
7080 RETURN

8000 DIM A(3,3)
8005 DIM B(3,3)
8010 DIM A$(2)
8020 DIM X(4)
8030 DIM Y(4)
8040 FOR Q=1 TO 3
8050 FOR W=1 TO 3
8060 A(Q,W)=0
8070 B(Q,W)=0
8080 NEXT W
8090 NEXT Q
8120 FIN=0
8130 DR=0

8140 PRINT CHR$(125)
8150 POKE 752,1
8190 POSITION 11,9:PRINT "<RRRSRRRSRRR>"
8200 POSITION 14,6:PRINT "|";
8210 POSITION 18,6:PRINT "|";
8220 POSITION 14,8:PRINT "|";
8230 POSITION 18,8:PRINT "|";
8240 POSITION 14,7:PRINT "|";
8250 POSITION 18,7:PRINT "|";
8260 POSITION 14,10:PRINT "|";
8270 POSITION 18,10:PRINT "|";
8280 POSITION 14,11:PRINT "|";
8290 POSITION 18,11:PRINT "|";
8300 POSITION 14,12:PRINT "|";
```

```
8310 POSITION 18,12:PRINT "|";
8320 POSITION 14,14:PRINT "|";
8350 POSITION 18,14:PRINT "|";
8360 POSITION 14,15:PRINT "|";
8370 POSITION 18,15:PRINT "|";
8380 POSITION 14,16:PRINT "|";
8390 POSITION 18,16:PRINT "|";
8400 POSITION 11,13:PRINT "<RRRSRRRSRRR>"
8410 RETURN


8500 SOUND 0,100,10,10
8510 FOR D=1 TO 100
8520 NEXT D
8530 SOUND 0,0,0,0
8540 RETURN


9000 POSITION 10,20:PRINT "Ich habe gewonnen"
9010 GOTO 9600
9100 POSITION 10,20:PRINT "UNENTSCHIEDEN!   "
9110 GOTO 9600
9500 POSITION 10,20:PRINT "Du hast gewonnen "
9600 POSITION 10,21
9602 PRINT "Noch ein Spiel J/N ";:INPUT A$
9610 IF A$="J" THEN RUN
9620 GRAPHICS 0
```

```
8330 POSITION 16,12:PRINT "|";
8340 POSITION 14,14:PRINT "|";
8350 POSITION 16,14:PRINT "|";
8360 POSITION 14,15:PRINT "|";
8370 POSITION 16,15:PRINT "|";
8380 POSITION 14,16:PRINT "|";
8390 POSITION 16,16:PRINT "|";
8400 POSITION 11,19:PRINT "ERGEBNISSE"
8410 RETURN

8500 SOUND 0,100,10,10
8510 FOR D=1 TO 100
8520 NEXT D
8530 SOUND 0,0,0,0
8540 RETURN

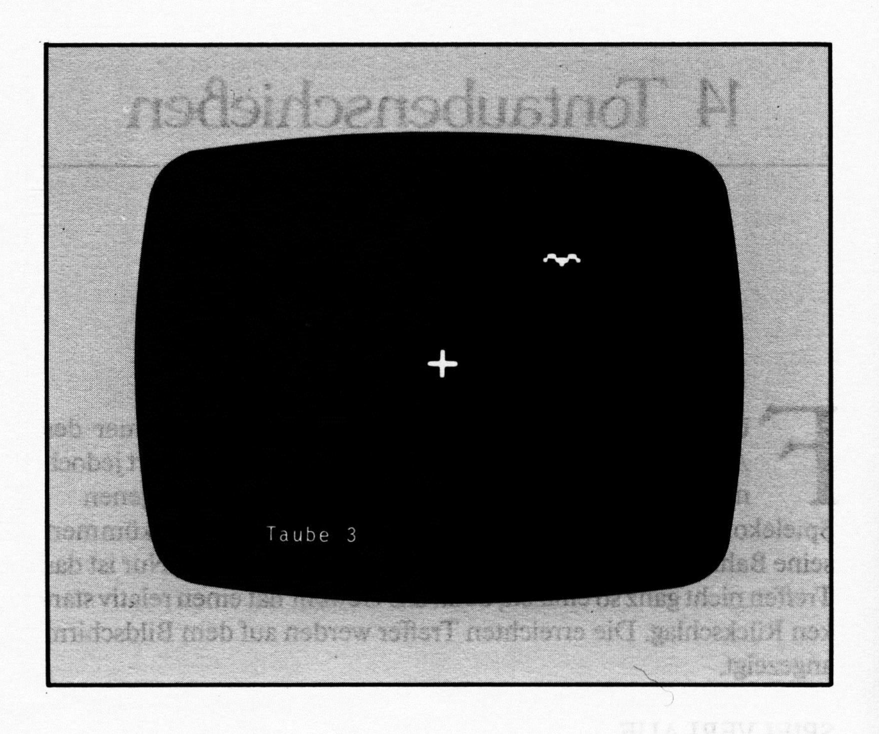
9000 POSITION 10,20:PRINT "Ich habe gewonnen"
9010 GOTO 9600
9100 POSITION 10,20:PRINT "UNENTSCHIEDEN"
9110 GOTO 9600
9500 POSITION 10,20:PRINT "Du hast gewonnen"
9600 POSITION 10,21
9602 PRINT "Noch ein Spiel J/N ";:INPUT A$
9610 IF A$="J" THEN RUN
9620 GRAPHICS 0
```

14 Tontaubenschießen

Fünf Tontauben stehen bereit, sich dem Gewehrfeuer des Atarischützen zu stellen. Ein eventueller Treffer führt jedoch nicht zum Absturz, sondern wird als Punkt dem eigenen Spielekonto zugeschlagen. Der künstliche Vogel zieht unbekümmert seine Bahn, unabhängig davon, wie oft er getroffen wird. Nur ist das Treffen nicht ganz so einfach, denn das Gewehr hat einen relativ starken Rückschlag. Die erreichten Treffer werden auf dem Bildschirm angezeigt.

SPIELVERLAUF

Das Fadenkreuz auf dem Bildschirm ist mit Hilfe der vier Pfeiltasten ins Ziel zu bringen. Geschossen wird durch Drücken der Leertaste. Bitte beachten: Der Rückschlag des Gewehrs muß durch entsprechendes Zielen ausgeglichen werden! Jede Taube darf so oft wie möglich getroffen werden.



PROGRAMMAUFBAU

20	Hauptprogramm
1000	Ausgabe des Fadenkreuzes auf dem Bildschirm
2000	Zielvorgang
3000	Anfangswerte, Grundeinstellungen des Atari für das Spiel
4000	Ausgabe der Taube auf dem Bildschirm
5000	Rückschlag beim Abschuß, Schuß, Test auf Treffer
6000	Tonerzeugung für Treffer
6500	Tonerzeugung für Schuß
8000	Titelbild und Berechnung der Flugbahn der Tontaube
9000	Spielende

ERLÄUTERUNGEN

Da es bei diesem Spiel auf möglichst kontinuierliche Bewegungen (Flug der Ton-Taube, Zielvorgang mit dem Fadenkreuz) ankommt, wird zur grafischen Darstellung die hochauflösende Grafik verwendet. Mit der LOCATE-Funktion wird im Unterprogramm ab Zeile 5000 geprüft, ob ein Treffer zu verzeichnen ist. Nachahmenswert ist das zur akustischen Untermalung des Schusses vorgesehene Unterprogramm ab Zeile 6500, das sich sehr gut zum Einbau in eigene Programme eignet. Um während des Spiels Rechenzeit zu sparen und die Abläufe zu beschleunigen, wird die Flugbahn der Taube vorher berechnet und für weitere Verwendungen im Feld B gespeichert.

```
10 REM Tontaubenschiessen

20 GOSUB 8000
30 FOR G=1 TO 5
35 PRINT CHR$(125)
38 GOSUB 8190
40 GOSUB 3000
45 PRINT "Taube ";G;
50 GOSUB 2000
60 NEXT G
70 GOTO 9000
90 STOP

1000 PLOT X-3,Y
1010 DRAWTO X+3,Y
1020 PLOT X,Y-3
1030 DRAWTO X,Y+3
1060 RETURN


2000 A=PEEK(764)
2005 POKE 764,255
2006 J=B(I):COLOR 0:GOSUB 4000
2008 COLOR 0:GOSUB 1000
2010 IF A=6 AND X>4 THEN X=X-2
2020 IF A=14 AND Y>4 THEN Y=Y-2
2030 IF A=15 AND Y<76 THEN Y=Y+2
2040 IF A=7 AND X<146 THEN X=X+2
2050 I=I+1:J=B(I):COLOR 2:GOSUB 4000
2060 IF A=33 THEN GOSUB 5000
2065 IF X<5 THEN X=5
2066 IF X>146 THEN X=146
2070 COLOR 1:GOSUB 1000
2080 IF FIN=1 THEN FIN=0:RETURN
2100 GOTO 2000

3000 FIN=0
3010 X=INT(RND(0)*30)+20
3020 Y=35
3030 I=6
3040 GOSUB 1000
3050 J=B(I):GOSUB 4000
3060 RETURN
```

```
4000 PLOT I,J-1
4005 PLOT I-3,J-2
4010 PLOT I-2,J-2
4020 PLOT I-1,J-1
4030 PLOT I,J
4040 PLOT I+1,J-1
4050 PLOT I+2,J-2
4060 PLOT I+3,J-2
4070 PLOT I-4,J-1
4080 PLOT I+4,J-1
4090 IF I>146 THEN FIN=1
4100 RETURN


5000 GOSUB 6500
5010 LOCATE X,Y,BI
5011 IF BI<>2 THEN X=X+10-INT(RND(0)*20):RETURN
5020 X=X+10-INT(RND(0)*20)
5030 GOSUB 6000
5040 PRINT " getroffen  ";
5050 H(G)=H(G)+1
5060 RETURN

6000 SOUND 0,50,10,8
6010 FOR D=1 TO 10
6020 NEXT D
6030 SOUND 0,0,0,0
6040 RETURN


6500 FOR P=15 TO 0 STEP -2
6510 SOUND 0,80,0,P
6515 FOR D=1 TO 15-P:NEXT D
6520 NEXT P
6530 SOUND 0,0,0,0
6540 RETURN

8000 HIT=0
8005 GRAPHICS 0
8010 POKE 752,1:PRINT CHR$(125)
8020 POSITION 12,2:PRINT "TONTAUBENSCHIESSEN"
8030 POSITION 10,10:PRINT "Einen Augenblick bitte"
8060 DIM B(160)
8070 FOR I=1 TO 160
8080 B(I)=(75-I)*(75-I)/150+10
8100 NEXT I
```

```
8130 DIM H(5)
8140 FOR H=1 TO 5
8150 H(H)=0
8160 NEXT H
8170 DIM A$(1)
8190 GRAPHICS 7
8200 POKE 752,1
8210 SETCOLOR 4,7,8
8230 SETCOLOR 1,0,14
8240 SETCOLOR 0,0,0
8250 SETCOLOR 2,7,8
8260 RETURN

9000 GRAPHICS 0
9010 T=0
9020 FOR I=1 TO 5
9030 POSITION 5,5+I
9040 PRINT "Taube ";I;" - Treffer: ";H(I)
9050 T=T+H(I)
9060 NEXT I
9070 POSITION 9,12:PRINT "Gesamttreffer: ";T
9080 POSITION 9,15:PRINT "Noch ein Spiel J/N ";
9090 INPUT A$
9100 IF A$="J" THEN RUN
9110 PRINT CHR$(125)
9120 POSITION 10,12
9130 PRINT "Auf Wiedersehen!"
```

15 Rettet den Wal

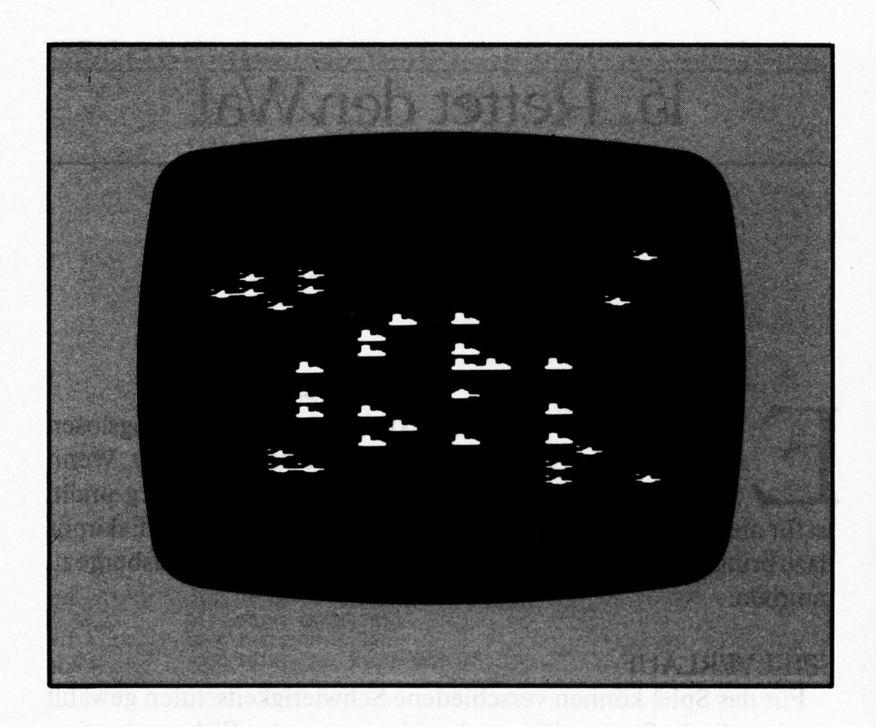
Einer der letzten Wale der Arktis wird von erbarmungslosen Eskimos gejagt. Eisberge bieten den einzigen Schutz. Wenn ein Eskimo mit seinem Kajak gegen einen Eisberg prallt, ist für diesen Jäger das Spiel zu Ende. Der Wal muß also die Eskimos dazu bringen, ihm zu folgen und im Eifer des Gefechts die Eisberge zu rammen.

SPIELVERLAUF

Für das Spiel können verschiedene Schwierigkeitsstufen gewählt werden, die die Startpositionen der schwimmenden Eisberge bestimmen. Das Verhalten des Wals wird über die Pfeiltasten gesteuert. Fährt ein Kajak gegen einen Eisberg, wird es zerstört. Rammt der Wal dagegen einen Eisberg, wird der Eisberg und damit ein möglicher Unterschlupf zerstört. Der Wal sollte also sehr gefühlvoll durch die arktischen Gewässer gesteuert werden. Das Spiel ist zu Ende, wenn ein Eskimo den Wal eingeholt und getötet hat oder wenn alle Eskimos an Eisbergen gescheitert sind.

PROGRAMMAUFBAU

20 Anfangswerte, Festlegen der Felder
160 Titelbild
250 Bildschirm für Grafik vorbereiten
280 Ausgabe der Kajaks auf dem Bildschirm



360	Ausgabe der Eisberge auf dem Bildschirm
420	Ausgabe des Wals auf dem Bildschirm
450	Hauptprogramm
520	Test auf Spielende
540	Bewegung des Wals
680	Bewegung der Kajaks
800	Spielende
950	Kopiert Zeichensatz in den RAM-Bereich
1000	Darstellung des Wals im RAM-Zeichensatz
1100	Darstellung eines Eskimos im RAM-Zeichensatz
1200	Darstellung eines Eisberges im RAM-Zeichensatz
1280	Umschalten auf den RAM-Zeichensatz und Abschalten des Cursors

1300 Farben
1600 Tonerzeugung

ERLÄUTERUNGEN

Die Anfangspositionen der Kajaks werden mit Hilfe der RND-Funktion innerhalb eines bestimmten Bereichs zufällig festgelegt (Zeilen 290 bis 300), ebenso die Startpositionen der Eisberge, allerdings unter Beachtung des Schwierigkeitsgrades (Zeilen 370 bis 380). Die LOCATE-Funktion in Zeile 735 prüft, ob ein Eisberg von einem Eskimo gerammt worden ist, was das Ende des Kajaks bedeuten würde. In gleicher Weise wird in Zeile 750 getestet, ob sich ein Eskimo dem Wal nähern und ihn erlegen konnte.

```
10 REM Rettet den Wal

20 GRAPHICS 0
30 OPEN #2,4,0,"K:"
40 DIM X(20)
50 DIM Y(20)
60 DIM U(20)
70 DIM V(20)

160 POSITION 12,2
161 PRINT "Rettet den Wal"
170 POSITION 4,7
171 PRINT "Eskimos in ihren Kajaks sind auf"
180 POSITION 4,9
181 PRINT "Walfang. - Der Wal wird mit Hilfe"
190 POSITION 4,11
191 PRINT "der Pfeiltasten gesteuert. Er"
200 POSITION 4,13
201 PRINT "kann nur entkommen, wenn es ihm"
202 POSITION 4,15
203 PRINT "gelingt, die Eskimos auf die"
204 POSITION 4,17
205 PRINT "Eisberge zu locken."
210 POSITION 1,20
211 PRINT "Schwierigkeitsgrad"
220 POSITION 1,22
221 PRINT "<1> schwer ... <3> leicht ";:GET #2,D
236 D=D-48
240 IF D<1 OR D>3 THEN GOTO 230

250 GRAPHICS 1+16:GOSUB 950

280 FOR C=1 TO 20
290 X=SGN(RND(0)-0.5)*INT(RND(0)*4+5)+9
300 Y=SGN(RND(0)-0.5)*INT(RND(0)*4+6)+9
310 POSITION X,Y:PRINT #6;CHR$(5);
320 X(C)=X
330 Y(C)=Y
340 NEXT C

360 FOR C=1 TO 20
370 U(C)=SGN(RND(0)-0.5)*INT(RND(0)*4+3-D)+9
380 V(C)=SGN(RND(0)-0.5)*INT(RND(0)*4+3-D)+9
390 POSITION U(C),V(C):PRINT #6;"@";
400 NEXT C
```

```
420 X=INT(RND(0)*2+10)
430 Y=INT(RND(0)*2+10)
440 POSITION X,Y:PRINT #6;CHR$(4+160)

450 GOSUB 540
460 F=0
470 FOR C=1 TO 20
480 IF X(C)=0 THEN GOTO 510
490 F=1
500 GOSUB 680
510 NEXT C

520 IF F=0 THEN GOTO 830
530 GOTO 450

540 GOSUB 1600
550 Z=X:V=Y
570 GET #2,A
580 IF A=0 THEN GOTO 570
590 IF A=43 AND X>1 THEN X=X-1
600 IF A=42 AND X<19 THEN X=X+1
610 IF A=61 AND Y<19 THEN Y=Y+1
620 IF A=45 AND Y>0 THEN Y=Y-1
640 POSITION Z,V:PRINT #6;" "
660 POSITION X,Y:PRINT #6;CHR$(4+160)
670 RETURN

680 POSITION X(C),Y(C):PRINT #6;" "
690 E=0
700 E=SGN(X(C)-X)
710 X(C)=INT(X(C)-E)
720 E=SGN(Y(C)-Y)
730 Y(C)=INT(Y(C)-E)
735 LOCATE X(C),Y(C),D
740 IF D=64 THEN X(C)=0:GOSUB 1600:GOTO 780
750 IF X(C)=X AND Y(C)=Y THEN GOTO 800
770 POSITION X(C),Y(C):PRINT #6;CHR$(5);
780 RETURN

800 POSITION X(C),Y(C):PRINT #6;" "
810 POSITION 1,15
811 PRINT #6;" der wal ist tot"
820 GOTO 840
```

```
830 POSITION 1,15
831 PRINT #6;"noch einmal entkommen"
840 DIM A$(1)
845 FOR Q=1 TO 500:NEXT Q
846 GRAPHICS 0
850 POSITION 10,10
851 PRINT "NOCH EIN SPIEL J/N ";
852 INPUT A$
860 IF A$="J" THEN RUN
870 GRAPHICS 0
900 STOP

950 CH=(PEEK(106)-8)*256
960 CHORG=PEEK(756)*256
970 FOR I=0 TO 511
980 POKE CH+I,PEEK(CHORG+I)
990 NEXT I

1000 POKE CH+(ASC("$")-32)*8+0,0
1010 POKE CH+(ASC("$")-32)*8+1,0
1020 POKE CH+(ASC("$")-32)*8+2,48
1030 POKE CH+(ASC("$")-32)*8+3,120
1040 POKE CH+(ASC("$")-32)*8+4,248
1050 POKE CH+(ASC("$")-32)*8+5,255
1060 POKE CH+(ASC("$")-32)*8+6,249
1070 POKE CH+(ASC("$")-32)*8+7,0

1100 POKE CH+(ASC("%")-32)*8+0,0
1110 POKE CH+(ASC("%")-32)*8+1,200
1120 POKE CH+(ASC("%")-32)*8+2,88
1130 POKE CH+(ASC("%")-32)*8+3,56
1140 POKE CH+(ASC("%")-32)*8+4,255
1150 POKE CH+(ASC("%")-32)*8+5,60
1160 POKE CH+(ASC("%")-32)*8+6,8
1170 POKE CH+(ASC("%")-32)*8+7,0

1200 POKE CH+(ASC("@")-32)*8+0,0
1210 POKE CH+(ASC("@")-32)*8+1,32
1220 POKE CH+(ASC("@")-32)*8+2,112
1230 POKE CH+(ASC("@")-32)*8+3,116
1240 POKE CH+(ASC("@")-32)*8+4,116
1250 POKE CH+(ASC("@")-32)*8+5,126
1260 POKE CH+(ASC("@")-32)*8+6,255
1270 POKE CH+(ASC("@")-32)*8+7,255
```

```
1280 POKE 756,CH/256
1290 POKE 752,1

1300 SETCOLOR 0,0,14
1310 SETCOLOR 1,3,4
1320 SETCOLOR 2,0,0
1340 SETCOLOR 4,7,8
1500 RETURN

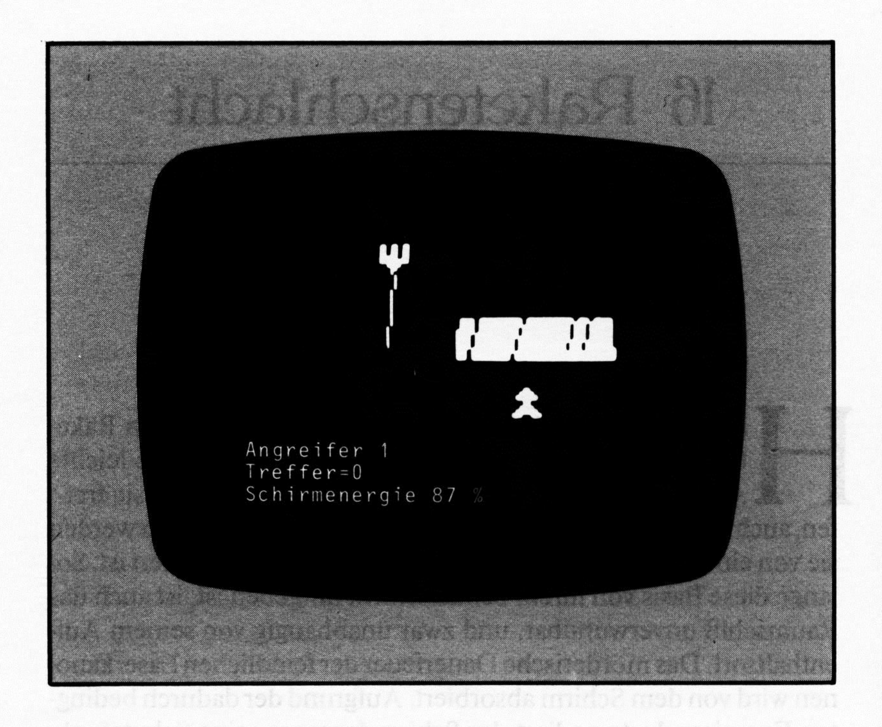
1600 SOUND 0,60,10,6
1610 FOR D=1 TO 50
1620 NEXT D
1630 SOUND 0,0,0,0
1640 RETURN
```

16 Raketenschlacht

Hier sind geniale Strategen herausgefordert! Mit zehn Raketen zehn Angreifer zu besiegen, ist bestimmt keine leichte Aufgabe. Die eigenen Raketen zerstören alles, was sie treffen, auch wenn es nur ein Streifschuß gewesen ist. Abgefeuert werden sie von einem Raumschiff, das in einer Raumbasis stationiert ist. Solange diese Basis von ihrem Schutzschirm umgeben ist, ist auch das Raumschiff unverwundbar, und zwar unabhängig von seinem Aufenthaltsort. Das mörderische Dauerfeuer der feindlichen Laserkanonen wird von dem Schirm absorbiert. Aufgrund der dadurch bedingten Energieverluste verliert der Schirm langsam seine Schutzfunktion. Bei 30% Restenergie ist er fast wirkungslos. Zur Kontrolle wird die jeweilige Reststärke des Schutzschirms angezeigt. Die Bahn eines gegnerischen Schiffs ist konstant und daher berechenbar. Trotzdem muß das Abfeuern einer Rakete genau überlegt werden. Es gibt nur zwei Punkte, von denen aus geschossen werden kann. Bei einem Fehlschuß ist die Rakete verloren, sie explodiert am oberen Bildschirmrand.

SPIELVERLAUF

Zum Schießen muß die Raumstation verlassen werden. Durch Betätigen der Rechts- oder Linkspfeiltaste nimmt das Schiff einen der beiden fest vorgegebenen Punkte ein und schießt eine Rakete ab. In die Kalkulation zum Verlassen der Basis müssen folgende Aspekte



einbezogen werden: Startposition und Flugzeit der Rakete sowie der Flugweg des gegnerischen Raumschiffs. Die gegnerischen Raumschiffe sollten möglichst früh abgeschossen werden, um das Dauerfeuer auf den Schutzschirm der Basis zu stoppen. Für die angreifenden Raumschiffe ist am oberen Bildschirmrand ein bestimmter Bereich vorgesehen. Innerhalb dieses Bereichs hat jedes Raumschiff eine eigene Geschwindigkeit und eine eigene Flugbahn.

HINWEISE

Vor und hinter dem Prozentzeichen in Zeile 690 (Angabe der Restenergie des Schutzschirmes) muß je ein Leerzeichen stehen. Wenn die Stärke des Schutzschirms 100% bzw. 10% unterschreitet, rückt das Prozentzeichen jedesmal um eine Stelle nach links, da die

Werte ganzzahlig sind (INT-Funktion) und wie üblich linksbündig ausgegeben werden. Um das alte Prozentzeichen zu löschen, ist das zweite Leerzeichen eingefügt worden.

PROGRAMMAUFBAU

30	Anfangswerte und Umschalten auf Grafik
70	Hauptprogramm
310	Ende des Spiels
380	Ausgabe des Gegners und dessen Dauerbeschuß auf dem Bildschirm
500	Test, ob eine Abwehrrakete abgeschossen werden soll
620	Berechnung der Restenergie des Schutzschirms
710	Abschuß der Abwehrrakete, Test auf Treffer
780	Explosion (Grafik und Ton)
940	Flugbahn des Gegners
1070	Ausgabe des Schutzschirmes auf dem Bildschirm
1200	Löschen des Bildschirms, Wahl der Farben
2000	Ausgabe der Explosion auf dem Bildschirm
2200	Löschen der alten Position der Rakete auf dem Bildschirm
2300	Ausgabe der Rakete auf dem Bildschirm
2400	Ausgabe des Gegners auf dem Bildschirm
2600	Tonerzeugung

ERLÄUTERUNGEN

Das Programm ist klar strukturiert, alle wichtigen Berechnungen und Bildschirmausgaben erfolgen durch Unterprogramme. Der Bereich, in dem sich ein gegnerisches Schiff aufhalten darf, wird nur einmal berechnet (ab Zeile 940) und in den Feldern X und Y gespeichert. Diese Werte stehen dann jederzeit sofort zur weiteren Verwendung zur Verfügung.

Das Durchlöchern des Schutzschirms ist relativ einfach darzustellen. Der gegnerische Laserstrahl ist in hochauflösender Grafik als eine einfache Linie in der Farbe 2 gezeichnet. Dieser Laserstrahl wird nun ein zweitesmal an gleicher Position in der Farbe 0 gezeichnet, wo-

durch der zuerst gezeichnete Strahl ausgelöscht wird. Bei diesem Löschvorgang wird nicht nur der Laserstrahl, sondern auch alles, was er durchdrungen hat, gelöscht. Auf dem Bildschirm ist das als Auflösung des Schutzschirms gut zu beobachten.

Die Restenergie des Schutzschirms wird entsprechend der gegnerischen Treffer berechnet (ab Zeile 620). Dazu werden mit Hilfe der LOCATE-Funktion die Treffer gezählt. LOCATE liefert genau dann den Wert 0, wenn der Punkt x, y die Farbe 0 hat, also ein Loch des Schirms darstellt.

```
10 REM Raketenschlacht

30 GOSUB 1200
40 GOSUB 940
50 GOSUB 1070
60 GOSUB 620

70 FOR A=1 TO 10
80 DIR=SGN(RND(0)-0.5)
90 PRINT "Angreifer ";A
100 PRINT "Treffer = ";HIT
110 R=7-INT(RND(0)*14)
120 IF R<0 THEN S=3-R:E=36
130 IF R>=0 THEN S=3:E=36-R
140 IF DIR=-1 THEN T=S:S=E:E=T
150 FOR I=S TO E STEP DIR
160 GOSUB 380
170 GOSUB 500
180 IF F=1 THEN GOSUB 710
190 NEXT I
200 GOSUB 2000
210 IF F=1 THEN FIN=1
220 IF FIN=2 THEN A=11:GOTO 300
230 GOSUB 620
240 IF F=1 THEN F=0:GOSUB 2100
250 GOSUB 2200
260 MX=74:MY=76
270 GOSUB 2300
280 IF FIN=0 THEN GOTO 150
290 FIN=0
300 NEXT A

310 IF FIN=2 THEN PRINT "Du wurdest abgeschossen"
315 IF FIN<2 THEN PRINT "Sehr gut!!!"
320 PRINT "Treffer = ";HIT
340 PRINT "Noch ein Spiel J/N ";
345 INPUT A$
350 IF A$="J" THEN RUN
360 GRAPHICS 0
370 STOP
```

```
380 REM Feind
390 GOSUB 2000
400 GOSUB 2400
410 IF RND(0)<0.3 THEN RETURN
420 HV=ABS(MX-X(I+DIR)+4)
421 IF C<=30 AND HV<10 THEN FIN=2:GOTO 840
430 COLOR 2:PLOT X(I+DIR)+4,Y(I+R+DIR)
440 D=5-INT(RND(0)*10)
450 DRAWTO X(I+DIR)+D,Y(I+R+DIR)+20
460 GOSUB 2600
470 COLOR 0:PLOT X(I+DIR)+4,Y(I+R+DIR)
480 DRAWTO X(I+DIR)+D,Y(I+R+DIR)+20
490 RETURN

500 REM Raketenfeuer
510 IF F=1 THEN RETURN
520 Z=PEEK(764)
530 POKE 764,255
540 IF Z=255 THEN RETURN
550 GOSUB 2200
560 IF Z=6 THEN MX=MX-40:GOTO 590
570 IF Z=7 THEN MX=MX+40:GOTO 590
580 RETURN
590 GOSUB 2300
600 F=1
610 RETURN

620 REM Schutzschirm
630 C=0
640 J=54
650 FOR I=60 TO 100
655 LOCATE I,J,P
660 C=C+P
670 NEXT I
680 C=INT(C/41*100)
690 PRINT "Schirmenergie: ";C;" % "
700 RETURN

710 REM Abwehrflak
720 GOSUB 2200
730 MY=MY-4
740 IF MY<12 THEN F=0:FIN=1:GOTO 860
750 GOSUB 2300
760 IF ABS(MX-X(I+DIR))>7 THEN RETURN
770 IF ABS(MY-Y(I+R+DIR))>5 THEN RETURN
```

```
780 FIN=1
790 GOSUB 2200
800 MX=X(I+DIR)
810 MY=Y(I+R+DIR)
820 HIT=HIT+1
830 F=0
840 COLOR 2:PLOT X(I+DIR)+3,Y(I+R+DIR)
850 DRAWTO MX+3,MY
860 GOSUB 2100
890 IF FIN=2 THEN GOTO 920
900 GOSUB 2000
910 X=X(I+DIR):Y=Y(I+R+DIR):COLOR 0:GOSUB 2430
920 I=E+DIR
930 RETURN


940 REM Orbit des gegnerischen Raumschiffs
950 DIM X(50),Y(50)
960 X=0:Y=0
970 N=39
980 FOR I=1 TO N
990 X=X+4
1000 Y=40-INT(((80-X)*(80-X))/200)
1010 X(I)=X
1020 Y(I)=Y
1030 NEXT I
1040 I=1
1050 HIT=0
1060 COLOR 1


1070 REM Schutzschild
1090 FOR I=50 TO 60
1100 PLOT 60,I:DRAWTO 100,I
1110 NEXT I
1120 MX=74
1130 MY=76
1160 GOSUB 2300
1170 F=0
1180 FIN=0
1190 RETURN


1200 REM Farben
1210 GRAPHICS 7
1220 SETCOLOR 4,8,8
1230 SETCOLOR 2,8,8
1240 SETCOLOR 1,0,14
1250 SETCOLOR 0,4,5
1260 DIM A$(1)
1320 RETURN
```

```
2000 COLOR 0
2010 X=X(I)
2020 Y=Y(I+R)
2030 GOTO 2430
2100 COLOR 2
2110 FOR Q=1 TO 20
2120 PLOT MX+INT(RND(0)*5),MY-INT(RND(0)*5)
2130 NEXT Q
2140 COLOR 0
2150 FOR Q=0 TO 7
2160 FOR Z=0 TO 7
2170 PLOT MX+Z,MY-Q
2180 NEXT Z
2190 NEXT Q
2195 RETURN

2200 COLOR 0
2210 GOTO 2310

2300 COLOR 1
2310 PLOT MX,MY:DRAWTO MX+2,MY
2320 PLOT MX+5,MY:DRAWTO MX+7,MY
2330 PLOT MX+1,MY-1:DRAWTO MX+6,MY-1
2340 PLOT MX+2,MY-2:DRAWTO MX+5,MY-2
2350 PLOT MX+3,MY-3:PLOT MX+4,MY-3
2360 PLOT MX+3,MY-4:PLOT MX+4,MY-3
2370 PLOT MX+1,MY-5:DRAWTO MX+6,MY-5
2380 PLOT MX+2,MY-6:DRAWTO MX+5,MY-6
2390 PLOT MX+3,MY-7:PLOT MX+4,MY-7
2395 RETURN

2400 COLOR 1
2410 X=X(I+DIR)
2420 Y=Y(I+R+DIR)
2430 PLOT X+3,Y:PLOT X+4,Y
2440 PLOT X+3,Y-1:PLOT X+4,Y-1
2450 PLOT X+2,Y-2:DRAWTO X+5,Y-2
2460 PLOT X,Y-3:DRAWTO X+7,Y-3
2470 PLOT X,Y-4:DRAWTO X+7,Y-3
2480 FOR Y1=Y-4 TO Y-7 STEP -1
2490 PLOT X,Y1
2500 PLOT X+1,Y1
2510 PLOT X+3,Y1
2520 PLOT X+4,Y1
2530 PLOT X+6,Y1
2540 PLOT X+7,Y1
2550 NEXT Y1
2560 RETURN
```

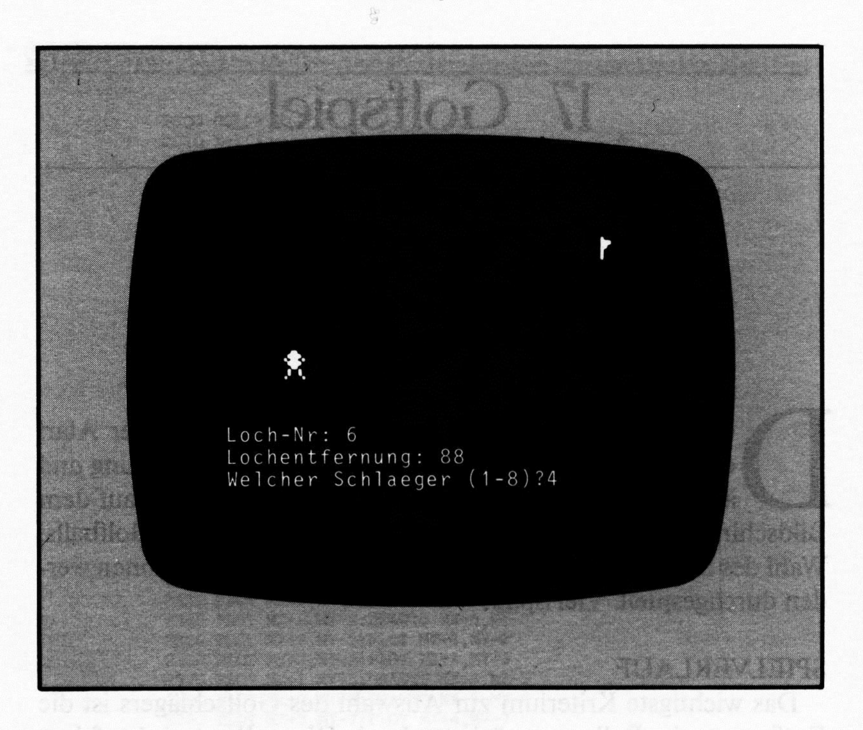
```
2600 SOUND 0,100,10,10
2610 FOR Q=1 TO 4
2620 NEXT Q
2630 SOUND 0,0,0,0
2640 RETURN
```

17 Golfspiel

Der Golfplatz ist zu weit entfernt? Macht nichts, der Atari ermöglicht mit beeindruckender grafischer Darstellung und äußerst realistischem Ablauf ein 9-Loch-Golfspiel auf dem Bildschirm. Das Wandern von Loch zu Loch, Putten des Golfballs, Wahl des Schlägers, Ballschlag, Flug des Balls, alle Situationen werden durchgespielt. Viel Spaß!

SPIELVERLAUF

Das wichtigste Kriterium zur Auswahl des Golfschlägers ist die Entfernung des Balls zum nächsten Loch. Dieser Wert wird auf dem Bildschirm angegeben. Acht verschiedene Golfschläger stehen zur Auswahl. Der Schläger mit der niedrigsten Nummer erzielt die größte Schlagweite. Fliegt der Ball über den Bildschirmrand, also aus dem Gelände, zählt das Loch als verfehlt, und es geht weiter bis zum nächsten. Wenn der Ball am Loch vorbeigeht, aber auf dem Platz bleibt, hat man einen neuen Versuch. Bei einem zu schwachen Schlag liegt der Ball etwas näher am Loch, von dort geht es weiter. Ist man dem Loch genügend nahe gekommen, wird der Ball auf dem letzten Stückchen regelrecht geputtet, bis er in das Loch gefallen ist. Der Spielstand wird laufend eingeblendet.



PROGRAMMAUFBAU

20	Anfangswerte der Variablen
2000	Nächstes Loch des Golfplatzes
4900	Punktzahl für jedes Loch
4930	Ende des Spiels
5000	Berechnung der Flugbahn des Balls
5500	Ausgabe des fliegenden Balls auf dem Bildschirm
6500	Test, ob der Ball am Loch vorbeigeflogen ist
7000	Schwingender Schläger beim Ballschlag
8000	Auswahl des Schlägers, Ausführung des Schlags
9000	Wahl der Farben
9200	Ausgabe des Fähnchens auf dem Bildschirm
9300	Ausgabe des Golfspielers auf dem Bildschirm

9400 Löschen der alten Position des Spielers auf dem Bildschirm schirm
9500 Tonerzeugung
9600 Erzeugung einer kurzen Pause
9700 Erzeugung einer langen Pause

ERLÄUTERUNGEN

Besondere Aufmerksamkeit verdient die Routine, mit der der Golfschläger bei dem bevorstehenden Ballschlag dargestellt wird (Unterprogramm ab Zeile 7000). Dazu wird in hochauflösender Grafik eine Linie gezeichnet und in einem zweiten Schritt sofort wieder gelöscht. Der Anfangspunkt der Linie ist fest in der Hand des Golfspielers, während sich der Endpunkt auf einem Kreis bewegt. Durch dieses fortwährendeZeichnen und Löschen der Linie in Verbindung mit der Bewegung des Endpunkts auf der Kreislinie entsteht der Eindruck, als ob mit dem Schläger wie im richtigen Spiel zum Schlag ausgeholt wird. Die Flugbahn des Balls, ebenfalls in hochauflösender Grafik, ist eine Parabel. Da der Ball immer in Richtung Loch fliegt, erscheint die Parabel etwas verzerrt.

EIGENE ERWEITERUNGEN

Trotz der schon vorhandenen umfangreichen Möglichkeiten kann das Spiel sinnvoll erweitert werden. So könnte jeder Ballschlag oder jedes Teilergebnis mit einer lustigen Meldung kommentiert werden. Wenn mehrere Spieler golfen, müßte für jeden Spieler eine Liste geführt und eine abschließende Rangfolgebewertung vorgenommen werden.

```
10 REM Golfspiel (9-Loch-Platz)

20 DIM A$(1)
790 DIM T(9)
810 XH=0:XC=0
820 YH=0:HT=0:YC=0

2000 FOR H=1 TO 9
2010 GOSUB 9000
2020 T(H)=0
2040 PRINT "Loch-Nr: ";H
2100 REM Lochdaten
2120 X=INT(RND(0)*25)+5
2130 Y=INT(RND(0)*10+60)
2150 XT=INT(RND(0)*60+85)
2170 YT=INT(RND(0)*30)+6
2180 D=SGN(XT-X)*
     SQR((XT-X)*(XT-X)+(YT-Y)*(YT-Y))
2190 GOSUB 9200
2200 GOSUB 9300
2300 PRINT "Lochentfernung: ";INT(D)
2310 PRINT "Welcher Schlaeger (1-8) ";
2311 INPUT C
2315 IF C<1 OR C>8 THEN GOTO 2310
2320 C=10-C
2325 GOSUB 7000
2326 B=0
2330 GOSUB 5000
2335 IF B=1 THEN B=0:GOTO 4920
2340 D=SGN(XT-XHT)*
     SQR((XT-XHT)*(XT-XHT)+(YT-YHT)*(YT-YHT))
2360 IF D<-10 THEN PRINT
     "zu weit - noch einmal":GOSUB 9700:GOTO 2010
2370 IF D<5 THEN PRINT
     "prima, schon nah am Ziel"
     :GOSUB 9700:GOTO 8000
2380 GOSUB 9400
2390 X=XHT
2400 Y=YHT
2500 GOTO 2190

4900 PRINT "Du brauchtest ";T(H);" Schlaege"
4910 GOSUB 9700
4920 NEXT H
```

```
4930 GRAPHICS 0
4940 POSITION 5,2:PRINT "Ergebnis dieser Runde"
4945 PRINT
4950 FOR I=1 TO 9
4960 PRINT "    Loch ";I;
4970 IF T(I)=-1 THEN PRINT " Ball verloren":GOTO 4990
4980 PRINT " ";T(I);" Schlaege"
4990 NEXT I
4995 PRINT :PRINT "Noch eine Runde J/N ";
4996 INPUT A$
4997 IF A$="J" THEN RUN
4999 GRAPHICS 0:STOP


5000 REM Schlagweite
5100 VT=(C*(1+RND(0)*0.1))+1
5160 HT=0
5170 XH=0


5500 REM Flug des Golfballs
5530 Q=(Y-YT)/(XT-X)
5600 VV=-VT*(SIN(45*3.14159/180))
5610 XC=X+1
5620 YC=Y
5630 VH=VT*(COS(45*3.14159/180))
5650 HT=HT+VV
5660 YH=Q*XH
5670 VV=VV+1
5800 XH=XH+VH
5810 YH=-Q*XH
5820 IF XH+XC>159 THEN YH=0:YT=0:YC=0:
     XH=0:XC=0:GOTO 6500
5830 IF YH+HT+YC<0 THEN YH=0:YT=0:YC=0:
     XH=0:XC=0:GOTO 6500
5850 IF HT>=0 THEN GOTO 5900
5860 COLOR 2:PLOT XH+XC,YH+HT+YC
5865 GOSUB 9600
5880 COLOR 0:PLOT XH+XC,YH+HT+YC
5890 GOTO 5650
5900 XHT=XH+XC
5910 YHT=YH+HT+YC
5915 RETURN
6000 PLOT XH+XC,YH+HT+YC
6010 RETURN
```

```
6500 PRINT "Der Ball ist weggeflogen"
6510 GOSUB 9500
6520 HT=0:T(H)=-1
6530 GOSUB 9700
6540 B=1:RETURN

7000 REM Schwingen des Schlaegers
7010 T(H)=T(H)+1
7100 XS=X+3
7110 YS=Y:YS=YS-3
7200 FOR S=-30 TO 5 STEP 2
7220 A=S/30*3.14159
7230 SX=6*SIN(A):SY=6*COS(A)
7240 COLOR 2:PLOT XS,YS:DRAWTO XS+SX,YS+SY
7250 IF S=0 THEN GOSUB 9500
7260 GOSUB 9600
7270 COLOR 0:PLOT XS,YS:DRAWTO XS+SX,YS+SY
7275 GOSUB 9300
7280 NEXT S
7500 RETURN

8000 REM putten des Golfballs
8010 GOSUB 9000
8030 XG=INT(RND(0)*15)+15
8040 YG=70
8050 XH=INT(RND(0)*45)+100
8060 YH=70
8070 D=XH-XG
8080 IF D<0 THEN D=ABS(D)
8100 XT=XH:YT=YH:GOSUB 9200
8110 X=XG:Y=YG:GOSUB 9300
8120 PRINT "Lochentfernung: ";D;"  "
8130 PRINT "Welcher Schlaeger (1-8) ";:INPUT C
8135 IF C<1 OR C>8 THEN GOTO 8130
8140 T(H)=T(H)+1
8145 H1=9-C+INT(RND(0)*2)
8150 H1=H1*5
8160 D=D-H1
8165 IF D<0 THEN H1=XH-XG
8170 FOR Z=XG+1 TO XG+H1
8180 COLOR 2:PLOT Z,YH
8200 COLOR 0:PLOT Z,YH
8210 NEXT Z
8220 GOSUB 9400
8230 XG=XH-D
8240 IF ABS(XG-XH)<4 THEN GOTO 8300
8250 IF D<0 THEN GOSUB 9000:XG=XH+2*D:GOTO 8070
8260 GOTO 8100
```

```
8300 REM im Loch !!!
8310 GOTO 4900

9000 GRAPHICS 7
9010 SETCOLOR 0,0,0
9020 SETCOLOR 4,12,9
9030 SETCOLOR 1,0,14
9040 SETCOLOR 2,12,9
9050 RETURN

9200 COLOR 1
9210 PLOT XT,YT
9220 DRAWTO XT,YT-5
9230 DRAWTO XT+2,YT-4
9240 DRAWTO XT,YT-3
9250 RETURN

9300 COLOR 2
9310 PLOT X,Y:PLOT X+5,Y
9320 PLOT X+1,Y-1:PLOT X+4,Y-1
9330 PLOT X+1,Y-2:PLOT X+4,Y-2
9340 PLOT X+2,Y-3:PLOT X+3,Y-3
9350 PLOT X+1,Y-4:DRAWTO X+4,Y-4
9360 PLOT X,Y-5:PLOT X+2,Y-5:
     PLOT X+3,Y-5:PLOT X+5,Y-5
9370 PLOT X+1,Y-6:DRAWTO X+4,Y-6
9380 PLOT X+2,Y-7:PLOT X+3,Y-7
9390 RETURN

9400 COLOR 0
9410 GOTO 9310

9500 SOUND 0,80,10,8
9510 GOSUB 9600
9520 SOUND 0,0,0,0
9530 RETURN

9600 FOR TON=1 TO 15
9610 NEXT TON
9620 RETURN

9700 FOR PAUSE=1 TO 500
9710 NEXT PAUSE
9720 RETURN
```

```
8300 REM im Loch !!!
8310 GOTO 4900

9000 GRAPHICS 7
9010 SETCOLOR 0,0,0
9020 SETCOLOR 4,12,2
9030 SETCOLOR 1,0,14
9040 SETCOLOR 2,12,9
9050 RETURN

9200 COLOR 1
9210 PLOT XT,YT
9220 DRAWTO XT,YT-5
9230 DRAWTO XT+2,YT-4
9240 DRAWTO XT,YT-3
9250 RETURN

9300 COLOR 2
9310 PLOT X,Y:PLOT X+5,Y
9320 PLOT X+1,Y-1:PLOT X+4,Y-1
9330 PLOT X+1,Y-2:PLOT X+4,Y-2
9340 PLOT X+2,Y-3:PLOT X+3,Y-3
9350 PLOT X+1,Y-4:DRAWTO X+4,Y-4
9360 PLOT X,Y-5:PLOT X+2,Y-5:
     PLOT X+3,Y-5:PLOT X+5,Y-5
9370 PLOT X+1,Y-6:DRAWTO X+4,Y-6
9380 PLOT X+2,Y-7:PLOT X+3,Y-7
9390 RETURN

9400 COLOR 0
9410 GOTO 9310

9500 SOUND 0,80,10,8
9510 GOSUB 9600
9520 SOUND 0,0,0,0
9530 RETURN

9600 FOR TON=1 TO 15
9610 NEXT TON
9620 RETURN

9700 FOR PAUSE=1 TO 500
9710 NEXT PAUSE
9720 RETURN
```

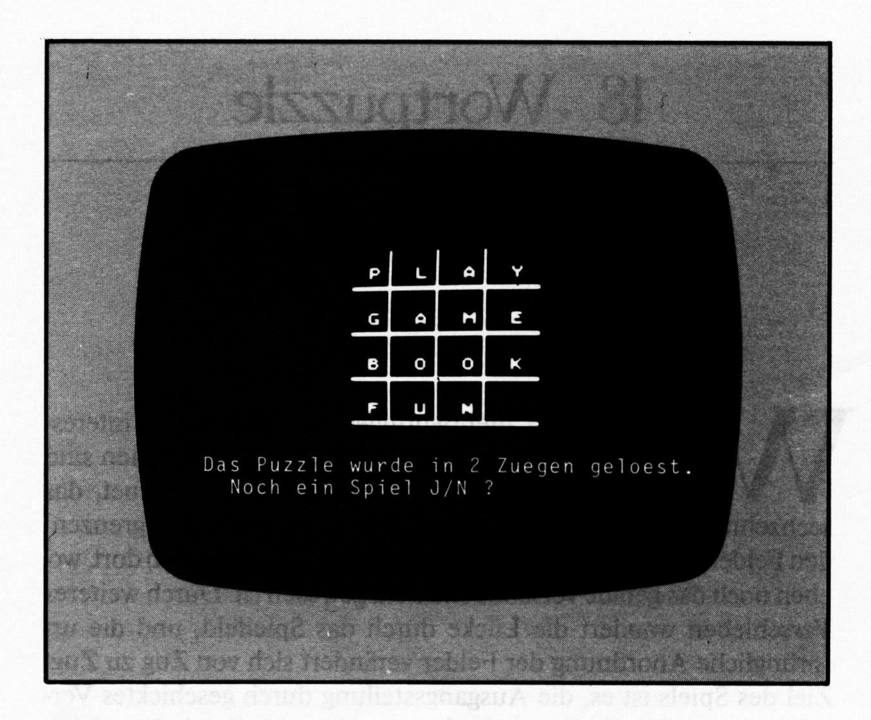
18 Wortpuzzle

Wortpuzzle ist ein altehrwürdiges, aber ungeheuer interessantes Spiel. In einem quadratischen 4 × 4-Rahmen sind fünfzehn frei verschiebbare Spielfelder angeordnet, das sechzehnte Feld bleibt frei. In diese Lücke kann eines der angrenzenden Felder verschoben werden. Die Lücke befindet sich nun dort, wo eben noch das gerade verschobene Feld gewesen ist. Durch weiteres Verschieben wandert die Lücke durch das Spielfeld, und die ursprüngliche Anordnung der Felder verändert sich von Zug zu Zug. Ziel des Spiels ist es, die Ausgangsstellung durch geschicktes Verschieben der Spielfelder wieder herzustellen. Als Symbole auf den Feldern werden Buchstaben verwendet. Auch für alte Hasen ist das Spiel eine Herausforderung!

SPIELVERLAUF

Es können nur direkt an die Lücke angrenzende Spielsteine bewegt werden, diagonale Züge sind nicht möglich. Die Felder sind von 1 bis 16 durchnumeriert. Der Zug wird ausgeführt, indem die Nummer des Feldes eingegeben wird, das in die Lücke geschoben werden soll. Fehlerhafte Eingaben werden als Hilflosigkeit gedeutet, und anstelle eines Spielzuges erscheinen für kurze Zeit die Feldnummern.

Mit welchen Buchstaben und Zeichen die fünfzehn Felder gefüllt werden sollen, kann zu Beginn des Spiels bestimmt werden. Einzugeben sind drei Wörter aus vier Buchstaben und ein Wort aus drei Buch-



staben. Durch geschickte Wahl dieser Wörter kann der Schwierigkeitsgrad des Spiels beeinflußt werden. Je mehr gemeinsame Buchstaben die Ratewörter besitzen, desto einfacher wird das Puzzle. Der höchste Schwierigkeitsgrad ergibt sich bei Wörtern, die keine gemeinsamen Buchstaben haben. Hat man keine eigenen Ideen, füllt der Computer die Felder mit den Buchstaben A bis O (das sind die ersten fünfzehn Buchstaben des Alphabets).

BEISPIELE:

niedriger Schwierigkeitsgrad	MOOR BOOT TRAM RAT
hoher Schwierigkeitsgrad	GOLD ACHT NEUN IST

Um eine gewisse «Unordnung» des Puzzles zu erreichen, werden die Felder nun durcheinander geschüttelt. Die Anzahl der vom Atari vorgenommenen Mischungen kann gewählt werden. Das Puzzle wird um so verzwickter, je länger der Computer neue Kombinationen errechnen kann. Das Puzzle ist gelöst, wenn durch geschicktes Verschieben der Spielfelder alle eingegebenen Ratewörter wieder im Klartext lesbar sind.

Und nun viel Erfolg beim puzzlen! Hier zeigen sich Ausdauer und Kombinationsgabe! Der Computer zählt mit und gibt am Schluß die Gesamtzahl der benötigten Spielzüge an.

HINWEISE

Den senkrechten Strich in den Zeilen 2020 und 2050 erreicht man durch gleichzeitiges Drücken der SHIFT- und der =-Taste (Gleichheitszeichen). Vor dem Strich der Zeile 2020 müssen zwei Leerzeichen stehen. Ebenfalls zwei Leerzeichen gehören zwischen die Anführungsstriche in Zeile 3250.

In diesem Programm werden einige der Grafikzeichen des Atari benötigt. Da diese Zeichen nicht im Programmlisting ausgedruckt werden können, sind sie in die spitzen Klammern < und > gesetzt (Zeilen 2080, 2130, 2180 bis 2220). In Zeile 2080 ist z.B. die Grafik-Buchstabenfolge RRS einzugeben, im Listing steht also <RRS>. Beim Erreichen der Klammer < ist der Atari zunächst in den Grafik-Modus zu schalten (Tasten CAPS und CTRL gemeinsam drücken), dann wird die Buchstabenfolge RRS eingetippt. Die Klammer > bedeutet Abschalten des Grafik-Modus (CPS und SHIFT gemeinsam drücken).

PROGRAMMAUFBAU

10 Definition der benötigten Felder
40 Auswahl verschiedener Optionen des Atari für das Spiel
120 Hauptprogramm
170 Ende des Spiels
500 Test, ob Ausgangszustand wieder erreicht ist

1000 Füllen der 15 Felder mit den Buchstaben A bis O
2000 Spielfeld
2500 Mischen der eingegebenen Wörter
3000 Einblenden der Feldnummern
3500 Löschen der Feldnummern, Fehlermeldungen
4000 Eingabe und Überprüfung des nächsten Zuges
5000 Feststellen, in welchem Feld sich die Lücke befindet
5500 Titelbild, einleitende Fragen
6000 Ausführen des Spielzugs
6500 Eingabe der Puzzlewörter
7000 Tonerzeugung
8000 Ausführen des Spielzuges (Fortsetzung)

ERLÄUTERUNGEN

Das Verschieben der Felder ist ziemlich kompliziert und benötigt umfangreiche Tests und Verzweigungen. Um das Programm dennoch überschaubar zu schreiben, ist wieder die bewährte Unterprogrammtechnik angewendet worden. Die Programme werden dadurch zwar länger, oft sogar langsamer, sind aber wesentlich leichter nachvollziehbar. Einzelschritte werden transparent, Fehlersuche und Programmänderungen erheblich vereinfacht.

EIGENE ERWEITERUNGEN

Nützlich wäre eine Routine, die die eingegebenen Wörter als Hilfestellung - z.B. als Liste - einblenden kann.

```
5 REM Wortpuzzle

10 DIM B(20)
20 DIM B$(20)
30 DIM W$(20)
35 DIM M$(5)
36 DIM A$(10)

40 GOSUB 5500
50 GOSUB 1000
60 IF WR=1 THEN GOSUB 6500
70 MOVE=0:ERROR=0
85 PRINT CHR$(125)
90 GOSUB 2000
100 GOSUB 5000
110 GOSUB 2500

120 GOSUB 4000
130 GOSUB 500
140 MOVE=MOVE+1
150 IF FIN<>0 THEN GOTO 120
160 POSITION 0,19
161 PRINT "Das Puzzle wurde in ";MOVE;
162 PRINT " Zuegen geloest."
170 PRINT :PRINT "Noch ein Spiel J/N ";
171 INPUT A$
180 IF A$="J" THEN RUN
200 PRINT CHR$(125)
210 POSITION 10,12

500 FIN=0:K=0
510 FOR I=1 TO 4
520 FOR J=1 TO 4
525 K=K+1
530 IF B$(B(I*4+J),B(I*4+J))<>W$(K,K) THEN FIN=1
540 NEXT J
550 NEXT I
560 RETURN

1000 K=0
1010 FOR I=1 TO 4
1020 FOR J=1 TO 4
1030 K=K+1
1040 B$(LEN(B$)+1)=CHR$(64+K)
```

```
1050 B(I*4+J)=K
1060 NEXT J
1070 NEXT I
1080 B$(16)=" "
1090 W$=B$
1100 RETURN

2000 FOR I=0 TO 3
2010 FOR J=0 TO 3
2020 POSITION 10+J*3,4+I*3:PRINT "  |";
2030 NEXT J
2040 FOR J=0 TO 3
2050 POSITION 10+J*3,5+I*3:PRINT " ";
2055 PRINT B$(B((I+1)*4+J+1),B((I+1)*4+J+1));
2056 PRINT "|";
2060 NEXT J
2070 FOR J=0 TO 3
2080 POSITION 10+J*3,6+I*3:PRINT "<RRS>";
2090 NEXT J
2100 NEXT I
2110 FOR I=0 TO 11
2120 POSITION 21,4+I:PRINT " ";
2130 IF INT((I+1)/3)=(I+1)/3 THEN GOTO 2150
2140 GOTO 2170
2150 POSITION 21,4+I:PRINT "<R>";
2170 NEXT I
2180 POSITION 12,15:PRINT "<X>";
2190 POSITION 15,15:PRINT "<X>";
2200 POSITION 18,15:PRINT "<X>"
2210 RETURN

2500 IS=4:JS=4:IY=0:JY=0
2510 FOR D=1 TO S
2520 I=IS:J=JS
2530 IF RND(0)>0.5 THEN GOTO 2570
2540 I=IS+INT(RND(0)*2)*2-1
2550 IF I>4 OR I<1 THEN I=IS:GOTO 2570
2560 GOTO 2590
2570 J=JS+INT(RND(0)*2)*2-1
2580 IF J>3 OR J<1 THEN J=JS:GOTO 2520
2590 IF I=IY AND J=JY THEN GOTO 2520
2600 IY=IS:JY=JS
2610 GOSUB 6000
2620 NEXT D
2630 RETURN
```

```
3000 K=0
3010 FOR I=0 TO 3
3020 FOR J=0 TO 3
3030 K=K+1
3040 POSITION 10+J*3,4+I*3:PRINT K
3050 NEXT J
3060 NEXT I
3070 RETURN

3500 FOR L=0 TO 3
3510 FOR P=0 TO 3
3520 POSITION 10+P*3,4+L*3:PRINT "  "
3530 NEXT P
3540 NEXT L
3555 IF ERROR=0 THEN RETURN
3560 POSITION 0,18
3570 FOR L=1 TO 40*5-4
3580 PRINT " ";
3590 NEXT L
3600 ERROR=0
3610 RETURN

4000 POSITION 2,19:PRINT "Bitte Zug eingeben: ";
4001 INPUT M$
4002 POSITION 2,19:PRINT "                    "
4003 REM ****************** ^ 20 Leerzeichen
4005 IF M$="" THEN GOTO 4000
4006 IF LEN(M$)>=2 THEN GOTO 4010
4007 A$(1)=M$(1):M$(1)="0":M$(2)=A$(1)
4010 IF M$(1,1)<"0" THEN GOTO 4017
4011 IF M$(1,1)>"9" THEN GOTO 4017
4012 IF M$(2,2)<"0" THEN GOTO 4017
4013 IF M$(2,2)>"9" THEN GOTO 4017
4015 GOTO 4020
4017 GOSUB 3000:GOTO 4000
4020 M=VAL(M$)
4030 IF M>0 AND M<17 THEN GOTO 4070
4035 ERROR=1
4040 POSITION 2,20
4041 PRINT "Als Zug muss eine Nummer von"
4050 POSITION 2,21
4051 PRINT "1 bis 16 eingegeben werden"
4060 GOSUB 3000
4065 GOTO 4000
4070 I=INT((M-1)/4)
4080 J=M-I*4
```

```
4090 I=I+1
4100 IF ABS(I-IS)+ABS(J-JS)=1 THEN GOTO 6000
4110 GOSUB 7000
4115 ERROR=1
4120 POSITION 2,20
4121 PRINT "Ein Spielstein kann nur in die"
4130 POSITION 2,21
4131 PRINT "Luecke geschoben werden.     "
4132 REM 5 Leerzeichen
4140 GOSUB 3000
4150 GOTO 4000

5000 K=0
5005 FOR L=0 TO 3
5010 FOR P=1 TO 4
5015 K=K+1
5020 IF B(P+L*3)=16 THEN IS=L:JS=P:MS=K
5030 NEXT P
5040 NEXT L
5050 RETURN

5500 PRINT CHR$(125)
5510 POSITION 10,1
5520 PRINT "W O R T P U Z Z L E"
5521 POSITION 10,2
5522 PRINT "==================="
5530 POSITION 6,10
5531 PRINT "Eigene Woerter eingeben J/N ";
5540 INPUT A$
5560 IF A$="J" THEN WR=1:GOTO 5600
5570 IF A$="N" THEN WR=0:GOTO 5600
5580 GOTO 5530
5600 POSITION 6,15
5605 PRINT "Wie oft Woerter mischen ";
5610 INPUT S
5620 IF S<1 THEN GOTO 5600
5640 RETURN

6000 GOSUB 3500
6070 GOSUB 8000
6080 POSITION 11+(J-1)*3,5+(I-1)*3
6081 PRINT B$(B(I*4+J),B(I*4+J));
6090 POSITION 11+(JS-1)*3,5+(IS-1)*3
6091 PRINT B$(B(IS*4+JS),B(IS*4+JS))
6140 RETURN
```

```
6500 PRINT CHR$(125)
6510 POSITION 2,4
6511 PRINT "Bitte 3 Woerter mit 4 Buchstaben"
6520 PRINT "und 1 Wort mit 3 Buchstaben eingeben."
6530 PRINT "1. Wort (4 Buchst.) ";:INPUT A$
6540 IF LEN(A$)<>4 THEN GOTO 6530
6550 W$=A$
6560 PRINT "Erstes Wort  = ";A$
6570 PRINT "2. Wort (4 Buchst.) ";:INPUT A$
6580 IF LEN(A$)<>4 THEN GOTO 6570
6590 W$(LEN(W$)+1)=A$
6600 PRINT "Zweites Wort = ";A$
6610 PRINT "3. Wort (4 Buchst.) ";:INPUT A$
6620 IF LEN(A$)<>4 THEN GOTO 6610
6630 W$(LEN(W$)+1)=A$
6640 PRINT "Drittes Wort = ";A$
6650 PRINT "4. Wort (3 Buchst.) ";:INPUT A$
6660 IF LEN(A$)<>3 THEN GOTO 6650
6670 W$(LEN(W$)+1)=A$
6680 PRINT "Viertes Wort = ";A$
6690 FOR Q=1 TO 100:NEXT Q
6700 W$(16)=" "
6710 B$=W$
6720 RETURN

7000 SOUND 0,80,10,8
7010 FOR F=1 TO 50
7020 NEXT F
7030 SOUND 0,0,0,0
7040 RETURN

8000 B(IS*4+JS)=B(I*4+J)
8030 B(I*4+J)=16
8040 T=IS:IS=I:I=T
8050 T=J:J=JS:JS=T
8060 RETURN
```



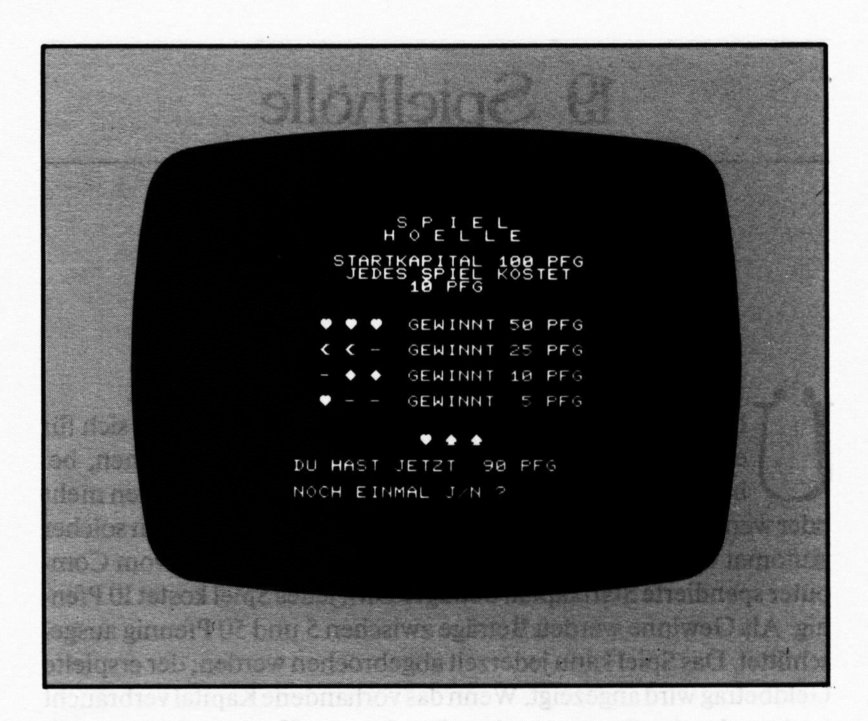
19 Spielhölle

Überall sind sie zu finden: Spielautomaten, bei denen sich für einen geringen Einsatz die Räder und Walzen drehen, bei bestimmten Kombinationen kann man sich über einen mehr oder weniger großen Gewinn freuen. Mit dem Atari kann ein solcher Automat vollkommen bargeldlos betrieben werden. Das vom Computer spendierte Startkapital beträgt 1 DM, jedes Spiel kostet 10 Pfennig. Als Gewinne werden Beträge zwischen 5 und 50 Pfennig ausgeschüttet. Das Spiel kann jederzeit abgebrochen werden; der erspielte Geldbetrag wird angezeigt. Wenn das vorhandene Kapital verbraucht ist, beendet der Computer als aufmerksamer Kassenwart das Spiel.

Obwohl das Programm nicht besonders lang ist, kommen die Freunde effektvoller grafischer Darstellung voll auf ihre Kosten. Als Spielsymbole laufen am Sichtfenster des Automaten Äpfel, Bananen, Kirschen und Glocken vorbei. Bei jedem Gewinn wird eine kurze Melodie gespielt.

SPIELVERLAUF

Die möglichen Gewinnkombinationen werden auf dem Bildschirm dargestellt. Ein Gewinn wird schon dann zugeteilt, wenn im Sichtfenster die Symbole einer Gewinnkombination in irgendeiner Reihenfolge auftauchen. Kombinationen wie Apfel-Apfel-Glocke und Apfel-Glocke-Apfel sind für die Gewinnentscheidung gleichwertig.



Leerzeichen in einer Gewinnkombination haben eine besondere Bedeutung. Es spielt dann keine Rolle, welches Symbol in dem entsprechenden Feld erscheint, es darf nur nicht mit den bereits vorgegebenen Gewinnsymbolen übereinstimmen. Ist z.B. als Gewinnkombination Herz-Leer-Herz angezeigt, gewinnen Kombinationen wie Herz-Pik-Herz oder Karo-Herz-Herz.

Das Programm wird mit RUN gestartet. Der Spielautomat muß mit «J» auf die entsprechende Frage hin in Gang gesetzt werden. Spieleinsatz und Gewinn werden automatisch verbucht. Beim vorzeitigen Beenden des Spiels wird der bis dahin angefallene Gewinn angezeigt.

PROGRAMMAUFBAU

20 Anfangswerte
50 Hauptprogramm
170 Symbole, mit denen das Spiel startet (Grundstellung der Walzen)
220 Drehen der Walzen
410 Test auf Gewinn, Buchung des gewonnenen Betrags
471 Eingerahmtes Titelbild
481 Kopiert Zeichensatz in den RAM-Bereich
540 Titelbild
630 Daten für die grafische Darstellung
740 Hauptgewinn
800 Spielabbruch, weil das gesamte Geld verbraucht ist
1000 Darstellung der benötigten Symbole im RAM-Zeichensatz
1060 Umschalten auf RAM-Zeichensatz, Abschalten des Cursors, Tastatur als Eingabefile eröffnen
2000 Tonerzeugung

ERLÄUTERUNGEN

Durch Anwendung effektvoller Tricks – wie sie die Programmiersprache BASIC ermöglicht – konnte das Programm trotz der zahlreichen grafischen Darstellungen und der Bewegungsabläufe erstaunlich kurz gehalten werden.

Die Symbole der Walzen werden in Punktreihen zerlegt (Matrixdarstellung). Jede Punktreihe wird als Element eines Feldes gespeichert. Der Effekt der drehenden Walzen wird folgendermaßen erreicht: Die oberste Punktreihe im Sichtfenster wird gelöscht. Alle übrigen Punktreihen werden nun um eine nach oben geschoben, und als letzte Zeile erscheint eine neue Punktreihe. So verschwindet ein Symbol langsam nach oben, während ein neues Symbol von unten nachgeschoben wird. Dieses Nachschieben geschieht einfach durch Erhöhen des Feldindexes.

EIGENE ERWEITERUNGEN

Der Automat könnte durch weitere Grafik zum «Einarmigen Banditen» vervollständigt werden. Weiter sollte das Drehen der Walzen und das Rasseln des Geldes bei der Gewinnauszahlung hörbar gemacht werden.

```
10 REM Spielhoelle

20 GOSUB 630
30 GOSUB 470
40 M=100

50 GOSUB 170
60 M=M-10
70 GOSUB 220
80 GOSUB 410
90 IF M<=0 THEN GOTO 800
100 POSITION 1,18
101 PRINT #6;"du hast jetzt"
102 POSITION 8,19
103 PRINT #6;" ";M;" PFG  "
110 POSITION 1,21
111 PRINT #6;"noch einmal j/n "
120 GET #2,A
130 IF A=32 THEN GOTO 120
140 IF A=74 THEN GOTO 50
150 POSITION 1,21
155 PRINT #6;"du kannst ";M;" PFG    "
156 REM *                        - 3 Leerz.
157 PRINT #6;" mitnehmen"
160 FOR I=1 TO 1000:NEXT I
165 STOP

170 REM Anfangswerte fuer Walzendrehung
180 X=INT(RND(0)*4)*10+1
190 Y=INT(RND(0)*4)*10+1
200 Z=INT(RND(0)*4)*10+1
210 RETURN

220 REM Walzendrehung
230 S=INT(RND(0)*2)+1
240 FOR I=0 TO S*10
250 FOR Q=0 TO 7
260 POKE CH+(ASC("$")-32)*8+Q,C(Q+X)
270 POKE CH+(ASC("%")-32)*8+Q,C(Q+X)
280 POKE CH+(ASC("&")-32)*8+Q,C(Q+X)
290 NEXT Q
310 POSITION 7,16:PRINT #6;"$ % &"
330 IF X=40 THEN X=0
```

```
340 IF Y=40 THEN Y=0
350 IF Z=40 THEN Z=0
360 X=X+1:Y=Y+1:Z=Z+1
370 NEXT I
380 X=X-1:Y=Y-1:Z=Z-1
390 RETURN

410 REM Bilanz nach einem Spiel
420 IF X=1 AND Y=1 AND Z=1 THEN GOSUB 740:RETURN
430 IF (X=11)+(Y=11)+(Z=11)=2 THEN M=M+25:GOSUB 2000
440 IF (X=21)+(Y=21)+(Z=21)=2 THEN M=M+10:GOSUB 2000
450 IF (X=1)+(Y=1)+(Z=1)=1 THEN M=M+5:GOSUB 2000
460 RETURN


470 GRAPHICS 1+16
471 FOR I=1 TO 20:PRINT #6;"*";:NEXT I
472 FOR I=1 TO 20
473 PRINT #6;"*";:POSITION 19,I:PRINT #6;"*";
474 NEXT I
475 POSITION 4,2:PRINT #6;"spielhoelle";
476 POSITION 3,12
477 PRINT #6;"gleich geht es";
478 POSITION 7,14:PRINT #6;"weiter"
479 POSITION 0,21
480 FOR I=1 TO 20:PRINT #6;"*";:NEXT I
481 CH=(PEEK(106)-8)*256
490 CHORG=(PEEK(756)*256)
500 FOR I=0 TO 511
510 POKE CH+I,PEEK(CHORG+I)
520 NEXT I
530 GOSUB 1000

540 GRAPHICS 1+16:POKE 756,CH/256
550 POSITION 5,0:PRINT #6;"s p i e l"
560 POSITION 4,1:PRINT #6;"h o e l l e"
570 POSITION 0,4:PRINT #6;"startkapital 100 PFG"
580 POSITION 0,6:PRINT #6;"jedes spiel kostet"
585 POSITION 14,7:PRINT #6;"10 PFG"
590 POSITION 0,9:PRINT #6;"! ! ! GEWINNT 50 PFG"
600 POSITION 0,10:PRINT #6;"@ @ - GEWINNT 25 PFG"
610 POSITION 0,11:PRINT #6;"- [ [ GEWINNT 10 PFG"
620 POSITION 0,12:PRINT #6;"! - - GEWINNT  5 PFG"
625 RETURN
```

```
630 DATA 06,10,20,36,68,207,239,230,0,0
640 DATA 02,12,28,56,56,28,12,02,0,0
650 DATA 24,60,60,60,126,255,24,24,0,0
660 DATA 12,24,124,255,255,255,126,60,0,0
670 DATA 06,10,20,36,68,207,239,230,0,0
680 DIM C(48)
690 FOR I=1 TO 48
700 READ C:C(I)=C
710 NEXT I
730 RETURN


740 REM Hauptgewinn
750 FOR I=50 TO 200 STEP 8
760 GOSUB 2000
770 NEXT I
780 M=M+50
790 RETURN

800 REM Geld verspielt
810 POSITION 0,21
815 PRINT #6;"das geld ist alle"
820 FOR I=1 TO 1000:NEXT I
825 STOP

1000 FOR Q=0 TO 7
1010 POKE CH+(ASC("$")-32)*8+Q,C(Q+1)
1015 POKE CH+(ASC("!")-32)*8+Q,C(Q+1)
1020 POKE CH+(ASC("%")-32)*8+Q,C(Q+10)
1025 POKE CH+(ASC("@")-32)*8+Q,C(Q+10)
1030 POKE CH+(ASC("&")-32)*8+Q,C(Q+20)
1035 POKE CH+(ASC("[")-32)*8+Q,C(Q+20)
1050 NEXT Q

1060 POKE 756,CH/256
1070 POKE 752,1
1080 OPEN #2,4,0,"K:"
1090 RETURN

2000 SOUND 0,80,10,8
2010 FOR T=1 TO 50
2030 NEXT T
2040 SOUND 0,0,0,0
2050 RETURN
```

```
630 DATA 06,10,20,36,68,207,239,230,0,0
640 DATA 02,12,28,56,58,28,12,02,0,0
650 DATA 24,60,60,60,126,255,24,24,0,0
660 DATA 12,24,124,255,255,255,126,60,0,0
670 DATA 06,10,20,36,68,207,239,230,0,0
680 DIM C(48)
690 FOR I=1 TO 48
700 READ C:C(I)=C
710 NEXT I
730 RETURN

740 REM Hauptgewinn
750 FOR I=50 TO 200 STEP 8
760 GOSUB 2000
770 NEXT I
780 M=M+50
790 RETURN

800 REM Geld verspielt
810 POSITION 0,22
815 PRINT #6;"das geld ist alle"
820 FOR I=1 TO 1000:NEXT I
825 STOP

1000 FOR Q=0 TO 7
1010 POKE CH+(ASC("*")-32)*8+Q,C(Q+1)
1015 POKE CH+(ASC("!")-32)*8+Q,C(Q+11)
1020 POKE CH+(ASC("&")-32)*8+Q,C(Q+1)
1025 POKE CH+(ASC("@")-32)*8+Q,C(Q+10)
1030 POKE CH+(ASC("%")-32)*8+Q,C(Q+20)
1035 POKE CH+(ASC("[")-32)*8+Q,C(Q+20)
1050 NEXT Q

1060 POKE 756,CH/256
1070 POKE 752,1
1080 OPEN #2,4,0,"K:"
1090 RETURN

2000 SOUND 0,50,10,8
2010 FOR T=1 TO 50
2030 NEXT T
2040 SOUND 0,0,0,0
2050 RETURN
```

20 Squash

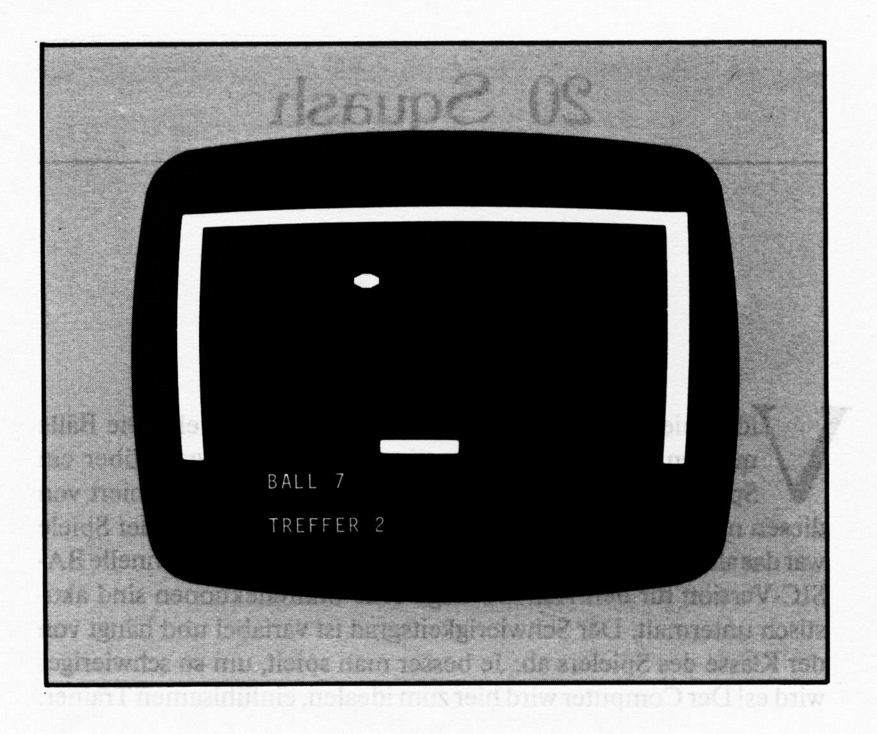
Videospiele der ersten Generation waren Ballspiele. Die Bälle mußten mit mehr oder weniger breiten Schlägern über ein Spielfeld gebracht werden, jung und alt waren fasziniert von diesen neuen Anwendungen des Fernsehapparats. Eines der Spiele war das als Squash bekannte 1-Mann-Tennis, das hier als schnelle BASIC-Version für den Atari vorliegt. Alle Ballreflektionen sind akustisch untermalt. Der Schwierigkeitsgrad ist variabel und hängt von der Klasse des Spielers ab: Je besser man spielt, um so schwieriger wird es! Der Computer wird hier zum idealen, einfühlsamen Trainer.

SPIELVERLAUF

Der Schläger befindet sich zu Beginn des Spiels in der Mitte am unteren Bildschirmrand, gesteuert wird er mit der Rechts- und Linkspfeiltaste. Bei jedem Treffer bewegt sich der Schläger ein Stückchen nach oben, was das Treffen des Balls außerordentlich erschwert. Umgekehrt wird das Spiel leichter, wenn der Ball nicht getroffen wird. Zehn Bälle gibt es, jeder Treffer wird gezählt. Der Spielstand ist während des gesamten Spiels sichtbar.

PROGRAMMAUFBAU

15 Initialisierung von Grafik und Variablen
150 Squash-Spielfeld und Anzahl der Bälle
200 Hauptprogramm



450 Einrahmung des Spielfelds
620 Ton der Ballreflektionen
770 Bewegung des Schlägers um ein Stück nach oben oder unten
890 Spielende und Ausgabe des Spielstandes
1010 Steuerung des Schlägers
1200 Löscht nicht mehr benötigtes Schlägerbild der vorhergehenden Darstellung (linke Seite)
1250 Löscht nicht mehr benötigtes Schlägerbild der vorhergehenden Darstellung (rechte Seite)
1500 Darstellung des Schlägers im RAM-Zeichensatz
1600 Darstellung des Balls im RAM-Zeichensatz
1700 Umschalten auf den RAM-Zeichensatz

2000 Tonerzeugung für Treffer
2500 Tonerzeugung für Ballverlust

ERLÄUTERUNGEN

Das Programm ist klar gegliedert; es dürfte keine Schwierigkeiten bereiten, die einzelnen Schritte zu erkennen und verstehen. Die Zeichen $ und % werden zur Darstellung von Schläger und Ball neu definiert. Zu diesem Zweck wird der Zeichensatz in den RAM-Bereich kopiert und entsprechend modifiziert. Die akustische Untermalung der Ballreflektionen (Unterprogramm ab Zeile 2000) kann abgeschaltet werden, indem die in Zeile 2000 stehenden Anweisungen durch RETURN ersetzt werden.

```
10 REM Squash

15 DIM B$(4)
20 GRAPHICS 1+16
21 POSITION 4,5
22 PRINT #6;"s q u a s h"
30 POSITION 2,13
31 PRINT #6;"einen augenblick"
32 POSITION 8,15
33 PRINT #6;"bitte"
40 CH=(PEEK(106)-8)*256
50 CHORG=(PEEK(756)*256)
60 FOR I=0 TO 511
70 POKE CH+I,PEEK(CHORG+I)
80 NEXT I
90 GOSUB 1500
95 PRINT #6;CHR$(125)
100 H=0
110 TH=0
120 D=19
130 BALL=0
140 C=2

150 GOSUB 450
190 X=10

200 BALL=BALL+1
210 IF BALL>10 THEN GOTO 890
220 A=10+INT(RND(0)*6)
230 B=2
240 V=1
250 W=1
260 Y=D
270 POSITION 3,21
280 PRINT #6;"BALL     ";
281 REM *       ^ 5 Leerzeichen
282 IF BALL<10 THEN PRINT #6;" ";
283 PRINT #6;BALL;
290 GOSUB 1010
300 POSITION X,Y:PRINT #6;B$
310 POSITION 3,22
320 PRINT #6;"TREFFER ";
321 IF HT<10 THEN PRINT #6;" ";
322 PRINT #6;HT
340 GOSUB 620
```

```
360 IF B+W<>Y THEN Y=D:GOTO 290
370 GOSUB 2500
380 POSITION X,Y
390 PRINT #6;"   ";:REM 3 Leerzeichen
400 GOSUB 620
410 POSITION A,B:PRINT #6;" "
420 IF D<19 THEN D=D+1
430 H=0
440 GOTO 200

450 REM Spielfeld
480 FOR I=0 TO 19
490 POSITION I,0:PRINT #6;"$";
500 NEXT I
510 FOR I=0 TO 19
520 POSITION 0,I
530 PRINT #6;"$";
540 POSITION 19,I
550 PRINT #6;"$";
560 NEXT I
570 RETURN


620 REM Ballreflektion
650 POSITION A,B
660 PRINT #6;" "
670 A=A+V
680 B=B+W
690 IF A=18 OR A=1 THEN V=-V:GOSUB 2000
700 IF B=1 THEN W=-W:GOSUB 2000
710 IF B+W=Y THEN GOTO 770
740 POSITION A,B
750 PRINT #6;CHR$(5+128)
760 RETURN


770 R=A-X
780 IF R<0 OR R>2 THEN GOTO 740
790 W=-W
800 GOSUB 2000
810 H=H+1
820 HT=HT+1
830 IF H<>1 THEN GOTO 720
840 H=0
850 IF D>3 THEN D=D-1
860 POSITION X,Y
870 PRINT #6;"   ";:REM 3 Leerzeichen
880 GOTO 760
```

```
890 FOR Z=1 TO 1000:NEXT Z
900 GRAPHICS 0:POSITION 10,10
910 PRINT "Ergebnis:"
911 POSITION 10,11
912 PRINT "---------"
915 POSITION 10,13
916 PRINT HT;" Treffer"
940 DIM A$(1)
950 POSITION 3,20
955 PRINT "Noch ein Spiel J/N ";:INPUT A$
970 IF A$="J" THEN RUN
980 PRINT ""
1000 END

1010 K=PEEK(764)
1040 IF K=7 AND X<16 THEN GOTO 1200
1050 IF K=6 AND X>1 THEN GOTO 1250
1060 RETURN

1200 POSITION X,Y
1210 PRINT #6;" ";
1220 X=X+1
1240 RETURN

1250 POSITION X+2,Y
1260 PRINT #6;" ";
1270 X=X-1
1290 RETURN

1500 FOR Q=0 TO 7
1510 POKE CH+(ASC("$")-32)*8+Q,255
1520 NEXT Q

1600 POKE CH+(ASC("%")-32)*8+0,60
1610 POKE CH+(ASC("%")-32)*8+1,126
1620 POKE CH+(ASC("%")-32)*8+2,255
1630 POKE CH+(ASC("%")-32)*8+3,255
1640 POKE CH+(ASC("%")-32)*8+4,255
1650 POKE CH+(ASC("%")-32)*8+5,255
1660 POKE CH+(ASC("%")-32)*8+6,126
1670 POKE CH+(ASC("%")-32)*8+7,60
```

```
1700 POKE 756,CH/256
1710 POKE 752,1
1720 B$="$$$"
1730 RETURN

2000 SOUND 0,80,10,8
2010 FOR T=1 TO 10:NEXT T
2020 SOUND 0,0,0,0
2030 RETURN

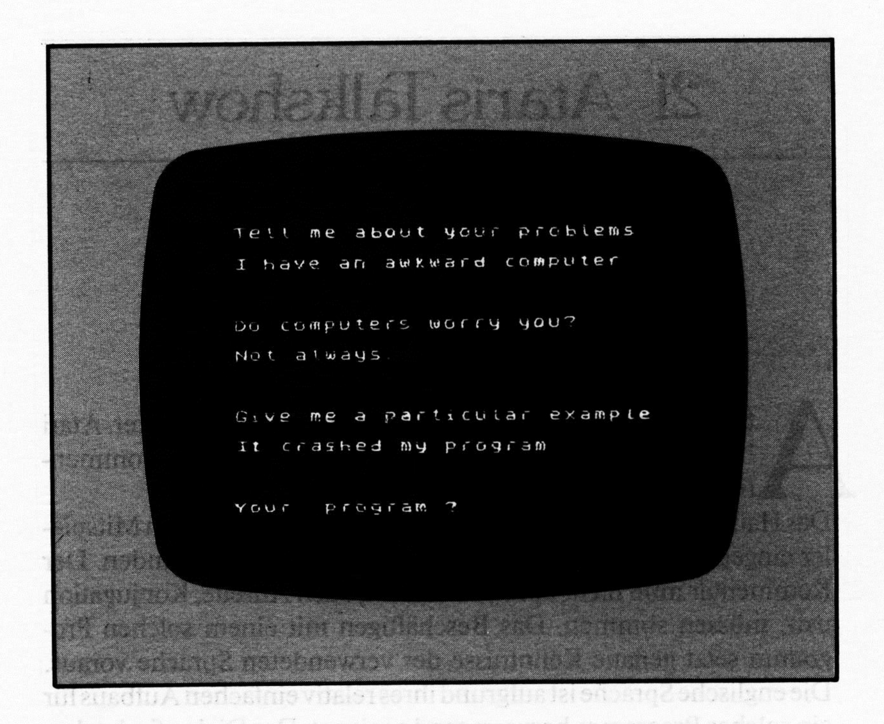
2500 FOR I=1 TO 10
2510 SOUND 0,160,10,10
2520 GOSUB 2000
2530 NEXT I
2540 RETURN
```

21 Ataris Talkshow

Ataris Talkshow ist eine Unterhaltung mit dem Computer. Atari hört sich geduldig Probleme und Wünsche an und kommentiert sie entsprechend.
Das Hauptproblem bei diesem Spiel liegt darin, zu dem vom Mitspieler eingegebenen Satz einen passenden Kommentar zu finden. Der Kommentar muß nicht nur sinnvoll sein, auch Anrede, Konjugation usw. müssen stimmen. Das Beschäftigen mit einem solchen Programm setzt genaue Kenntnisse der verwendeten Sprache voraus. Die englische Sprache ist aufgrund ihres relativ einfachen Aufbaus für ein solches Programm hervorragend geeignet. Der Dialog findet deshalb in Englisch statt. Es wird weiter deutlich, nach welchen Gesichtspunkten Antworten, auch im rein menschlichen Dialog, gegeben werden. Und nun viel Spaß bei «Ataris Talk-Show»!

SPIELVERLAUF

Atari eröffnet alle Talkshows mit der Frage nach irgendwelchen Problemen. Jede Antwort ist möglich, nach kurzer Denkpause hat der Computer eine passende Antwort gefunden, die er über den Bildschirm ausgibt. Die Eingaben an den Computer sollten mehr sein als ein einfaches «yes» oder «no», aber nicht über eine Zeile hinausgehen. Beim Beachten dieser kleinen Regel sind lange interessante Unterhaltungen möglich.



HINWEISE

Beim Eingeben des Programms bzw. der Antworten ist unbedingt auf korrekte Schreibweise zu achten! Der Computer führt keine orthografischen Tests durch. Bei Tippfehlern wird er daher keine Schlüsselwörter finden, nach denen er eine sinnvolle Antwort auswählen kann. Als einziges Satzzeichen ist der Apostroph erlaubt. Bitte in den Eingabesätzen keine Punkte oder Kommas verwenden.

«Ataris Talkshow» ist das längste Programm dieser Sammlung. Es läuft auf einem 16K-Atari nur, wenn die Diskettenlaufwerke abgeschaltet sind. Zum Abspeichern bitte einen Kassettenrecorder verwenden.

PROGRAMMAUFBAU

20 Hauptprogramm
1000 Begrüßung und Erläuterung des Spiels
2000 Antworten des Mitspielers
3000 Zerlegung der eingegebenen Sätze in Einzelwörter
3600 Aufbereitung der Computerantworten durch Änderung von Konjugation und Pronomen der Eingabe
3800 Paare von Konjugationen und Pronomen, wie sie in der Eingabe vorkommen können und für die Ausgabe benötigt werden
5000 Suche nach Schlüsselwörtern im Eingabetext
6000 Sammlung von Schlüsselwörtern
7000 Antworten zu den Schlüsselwörtern
9800 Ausgabe der Computerantwort auf dem Bildschirm
9900 Aufforderung, sinnvolle Sätze einzugeben

ERLÄUTERUNGEN

Jeder eingegebene Satz wird zunächst in Einzelwörter zerlegt. Anschließend überprüft der Computer, ob Schlüsselwörter im Eingabesatz vorkommen und stellt die entsprechende Antwort zusammen. Das Schlüsselwort «why» z.B. veranlaßt den Atari, als Antwort «Some questions are difficult to answer» auszugeben. Wenn ein Eingabesatz keines der vorgesehenen Schlüsselwörter enthält, wird einer der allgemeingültigen Sätze über die RND-Funktion ausgewählt.

Das alles hört sich zwar einfach an, es kostet den Computer aber schon einige Mühe, aus einem Eingabesatz die Antwort zusammenzubauen. Konjugation, Deklination, Wahl der Pronomen, alles soll stimmen! Entsprechend schwierig ist es, ein solches Programm zu schreiben oder zu modifizieren. Obwohl das diesem Programm zugrunde liegende Prinzip nicht sehr kompliziert ist, sind die Antworten vielfältig und erstaunlich gut auf die Eingabe abgestimmt.

EIGENE ERWEITERUNGEN

Sollen weitere Schlüsselwörter eingefügt werden, ist der Aufbau des Unterprogramms ab Zeile 6000 genauestens zu beachten. Jedem Schlüsselwort ist eine Zeilennummer zugeordnet, in der die zugehörige Antwort steht. Bei einem Eingabesatz mit mehreren Schlüsselwörtern ist das erste gefundene Schlüsselwort für den Computer maßgebend.

```
10 REM Ataris Talkshow

20 GOSUB 1000
30 GOSUB 2000
40 GOSUB 3000
50 GOSUB 5000
60 IF NUM<>0 THEN GOSUB NUM
70 GOSUB 9800
100 GOTO 30


1000 PRINT CHR$(125)
1005 OPEN #2,4,0,"K:"
1010 POSITION 2,1:PRINT "Hi there!"
1030 PRINT
1040 PRINT "I would like you to talk to me"
1050 PRINT "but I don´t have ears so will"
1060 PRINT "you type sentences on my"
1070 PRINT "keyboard."
1074 PRINT
1075 PRINT "Don´t use any punctuation apart"
1076 PRINT "from apostrophies which are"
1078 PRINT "important."
1080 PRINT
1110 PRINT "When you have finished typing"
1115 PRINT "press RETURN"
1120 POSITION 2,18
1121 PRINT "Tell me about your problems"
1125 DIM P$(100),R$(50),M$(50),D$(50),C$(50)
1130 R$=""
1140 M$=""
1150 D$=""
1160 DIM N$(3*32)
1165 FOR I=1 TO 3*32:N$(I)=" ":NEXT I
1170 N$(1,32)="Please go on"
1180 N$(33,64)="I´m not sure I understand you"
1190 N$(65,96)="Tell me more"
1200 DIM I$(3*40)
1205 FOR I=1 TO 3*32:I$(I)=" ":NEXT I
1210 I$(1,40)="Let´s talk some more about your"
1220 I$(41,80)="Earlier you spoke of your "
1230 I$(81,120)="Does that have anything to do with your "
1235 DIM J$(2*32)
1236 FOR I=1 TO 2*32:J$(I)=" ":NEXT I
1240 J$(1,32)="Are you just being negative"
1250 J$(33,64)="I see"
1260 DIM A$(100),B$(100),Z$(100)
1280 DIM T$(50)
1290 DIM W(20,2)
1990 RETURN
```

```
2000 A$=" "
2005 POKE 752,1
2010 GET #2,B
2030 IF B=155 THEN GOTO 2200
2035 IF B=126 AND LEN(A$)=1 THEN A$=" ":GOTO 2090
2040 IF B=126 AND LEN(A$)>1 THEN
     A$=A$(1,LEN(A$)-1):GOTO 2090
2050 IF B<32 OR B>123 THEN GOTO 2000
2060 A$(LEN(A$)+1)=CHR$(B)
2090 POSITION 2,20:PRINT A$;" ";
2100 GOTO 2010
2200 IF A$=" " THEN GOTO 2000
2201 IF A$(LEN(A$),LEN(A$))=" " THEN
     B$=A$(1,LEN(A$)-1):A$=B$:GOTO 2200
2202 IF A$=R$ THEN POSITION 0,21:PRINT
     "You´re repeating yourself!":
     PRINT :PRINT :GOTO 2000
2203 R$=A$
2204 IF A$=" " THEN GOTO 2000
2215 IF ASC(A$(2,2))<97 THEN A$(2,2)=CHR$(ASC(A$(2,2))+32)
2220 POSITION 0,21:PRINT
2230 RETURN


3000 FOR I=1 TO 20
3002 W(I,1)=0:W(I,2)=0
3004 NEXT I
3005 N=1
3010 B=0
3020 FOR I=1 TO LEN(A$)
3040 IF (A$(I,I)=" " OR A$(I,I)=",") AND B=0 THEN B=1
3050 IF (A$(I,I)<>" " AND A$(I,I)<>",") AND B<=1 THEN
     W(N,1)=I:B=2
3060 IF (A$(I,I)=" " OR A$(I,I)=",") AND B=2 THEN
     W(N,2)=I-1:N=N+1:B=0
3070 NEXT I
3080 W(N,2)=LEN(A$)
3085 A$(LEN(A$)+1)="   ":REM 3 Leerzeichen


3600 FOR I=1 TO N
3605 RESTORE 3800
3610 READ B$
3620 IF B$="s" THEN GOTO 3690
3630 IF B$<>A$(W(I,1),W(I,2)) THEN GOTO 3680
3640 READ C$
3650 Z$=A$(1,W(I,1)-1):Z$(LEN(Z$)+1)=C$:
     Z$(LEN(Z$)+1)=A$(W(I,2)+1)
```

```
3654 A$=Z$
3655 W(I,2)=W(I,2)+LEN(C$)-LEN(B$)
3656 FOR J=I+1 TO N
3660 W(J,2)=W(J,2)+LEN(C$)-LEN(B$)
3664 W(J,1)=W(J,1)+LEN(C$)-LEN(B$)
3665 NEXT J
3670 GOTO 3690
3680 READ B$
3690 NEXT I
3700 RETURN

3800 DATA my,your*,I,you*,i,you*
3810 DATA mum,Mother,dad,Father
3820 DATA dreams,dream,you,I*,me,you*
3830 DATA your,my*,myself,yourself*,Im,you´re
3840 DATA yourself,myself*,I´m,you´re*
3850 DATA you´re,I´m*,am,are*
3870 DATA I´m,you´re*
3880 DATA were,was
3885 DATA are,am
3890 DATA s,s

5000 RESTORE 6000
5010 READ B$,NUM
5020 IF B$="s" THEN GOTO 5710
5025 FOR I=1 TO N
5030 IF A$(W(I,1),W(I,2))<>B$ THEN GOTO 5700
5040 T$=A$(W(I,2)+1)
5050 RETURN
5700 NEXT I
5705 GOTO 5010
5710 NUM=0
5720 IF M$<>"" THEN GOTO 5800
5730 Z=INT(RND(0)*3):P$=N$(Z*32+1,Z*32+32)
5740 RETURN
5800 Z=INT(RND(0)*3):P$=I$(Z*32+1,X*32+32)
5805 P$(LEN(P$)+1)=M$
5900 RETURN

6000 DATA computer,7000,machine,7000
6010 DATA like,7100,same,7100,alike,7100
6020 DATA if,7200,everybody,7300
6024 DATA can,8200,certainly,8250
6025 DATA how,8100,because,8150
6026 DATA always,7800
```

```
6030 DATA everyone,7300,nobody,7300
6034 DATA was,7500
6035 DATA I*,8800
6040 DATA no,7400
6060 DATA your*,7600
6070 DATA you're*,8500,you*,8650
6110 DATA hello,8300,maybe,8350
6120 DATA my*,8370,no,8420
6130 DATA yes,8250,why,8450
6140 DATA perhaps,8350,sorry,8400
6160 DATA what,8450
6900 DATA s,0


7000 P$="Do computers worry you?"
7010 RETURN
7100 P$="In what way ?"
7110 RETURN
7200 P$="Why talk of possibilities"
7210 RETURN
7300 P$="Really "
7305 P$(LEN(P$)+1)=B$
7306 P$(LEN(P$)+1)=" ?"
7310 RETURN
7400 IF I=N THEN GOTO 7450
7410 I=I+1
7420 IF A$(W(I,1),W(I,2))="one"THEN
     B$(LEN(B$)+1)=" one":GOTO 7300
7450 Z=INT(RND(0)*2):P$=J$(Z*32+1,Z*32+32)
7460 RETURN
7500 IF I=N THEN GOTO 9900
7510 I=I+1
7515 IF I>N THEN GOTO 5720
7520 IF A$(W(I,1),W(I,2))<>"you*" THEN GOTO 7550
7530 P$="What if you were "
7535 P$(LEN(P$)+1)=A$(W(I,2)+1)
7536 P$(LEN(P$)+1)=" ?"
7540 RETURN
7550 IF A$(W(I,1),W(I,2))<>"I*" THEN GOTO 5720
7560 P$="Would you like to believe I was "
7565 P$(LEN(P$)+1)=A$(W(I,2)+1)
7570 RETURN
7600 I=I+1
7605 IF I>N THEN GOTO 7450
7610 IF A$(W(I,1),W(I,2))="Mother" THEN GOTO 7700
7620 IF A$(W(I,1),W(I,2))="Father" THEN GOTO 7700
7630 IF A$(W(I,1),W(I,2))="sister" THEN GOTO 7700
7640 IF A$(W(I,1),W(I,2))="brother" THEN GOTO 7700
```

```
7650 IF A$(W(I,1),W(I,2))="wife" THEN GOTO 7700
7660 IF A$(W(I,1),W(I,2))="husband" THEN GOTO 7700
7670 IF A$(W(I,1),W(I,2))="children" THEN GOTO 7700
7680 IF LEN(T$)>10 THEN M$=T$
7690 P$="your "
7692 P$(LEN(P$)+1)=T$
7694 P$(LEN(P$)+1)=" ?"
7695 RETURN
7700 P$="Tell me more about your family"
7710 RETURN
7800 P$="Give me a particular example"
7810 RETURN
7900 I=I+1
7905 IF I>N THEN P$="Am I what ?":RETURN
7920 P$="Why are you interested in whether I am "
7924 P$(LEN(P$)+1)=A$(W(I,1))
7926 P$(LEN(P$)+1)=" or not?"
7930 RETURN
8000 P$="Do you think you are "
8010 P$(LEN(P$)+1)=A$(W(I,2))
8030 RETURN
8100 P$="Why do you ask ?"
8110 RETURN
8150 P$="Tell me about any other reasons"
8160 RETURN


8200 I=I+1
8205 IF I>N THEN P$="What ?":RETURN
8210 IF A$(W(I,1),W(I,2))="I*" THEN
     P$="Do you believe I can":
     P$(LEN(P$)+1)=A$(W(I,2)+1):RETURN
8220 IF A$(W(I,1),W(I,2))="you*" THEN
     P$="Do you believe you can ":
     P$(LEN(P$)+1)=A$(W(I,2)+1):RETURN
8230 GOTO 5720
8250 P$="You seem very positive"
8260 RETURN
8300 P$="Pleased to meet you - let´s talk"
     about your problems"
8310 RETURN
8350 P$="Could you try to be more positive"
8360 RETURN
8370 P$="Why are you concerned about my"
8375 P$(LEN(P$)+1)=T$
8380 RETURN
8400 P$="you don´t have to apologise to me"
8410 RETURN
8450 P$="Some questions are difficult to answer ..."
```

```
8460 RETURN
8500 I=I+1
8505 IF I>N THEN GOTO 5720
8510 P$="I´m sorry to hear that you are":
     P$(LEN(P$)+1)=A$(W(I,1),W(I,2))
8520 IF A$(W(I,1),W(I,2))="sad" THEN RETURN
8530 IF A$(W(I,1),W(I,2))="unhappy" THEN RETURN
8540 IF A$(W(I,1),W(I,2))="depressed" THEN RETURN
8550 IF A$(W(I,1),W(I,2))="sick" THEN RETURN
8560 P$="How have I helped you to be ":
     P$(LEN(P$)+1)=A$(W(I,1),W(I,2))
8570 IF A$(W(I,1),W(I,2))="happy" THEN RETURN
8580 IF A$(W(I,1),W(I,2))="elated" THEN RETURN
8590 IF A$(W(I,1),W(I,2))="glad" THEN RETURN
8600 IF A$(W(I,1),W(I,2))="better" THEN RETURN
8610 P$="Is it because you are "
8615 P$(LEN(P$)+1)=A$(W(I,1))
8616 P$(LEN(P$)+1)=" you would like to talk to me"
8620 RETURN


8650 IF I=1 THEN GOTO 8655
8654 IF A$(W(I-1,1),W(I-1,2))="are*" THEN GOTO 8000
8655 I=I+1
8656 IF I>N THEN GOTO 9900
8660 IF A$(W(I,1),W(I,2))="are*" THEN GOTO 8500
8670 P$="What would it mean if you got ":
     P$(LEN(P$)+1)=A$(W(I,2)+1)
8675 IF A$(W(I,1),W(I,2))="want" OR
     A$(W(I,1),W(I,2))="need" THEN RETURN
8680 P$="How do you know you can´t":
     P$(LEN(P$)+1)=A$(W(I,2)+1)
8685 IF A$(W(I,1),W(I,2))="can´t" OR
     A$(W(I,1),W(I,2))="cannot" THEN RETURN
8690 IF A$(W(I,1),W(I,2))="feel" THEN
     P$="tell me more about how you feel":RETURN
8700 GOTO 5720
8800 IF I-1<1 THEN GOTO 8805
8804 IF A$(W(I,1),W(I,2))="am" THEN GOTO 7790
8805 I=I+1
8810 IF I>=N THEN P$="What am I ?":RETURN
8820 IF A$(W(I,1),W(I,2))="am" THEN
     P$="Why do you think so ?":RETURN
8830 P$="Is that what you think of me !"
8840 RETURN
```

```
9800 FOR J=1 TO LEN(P$)
9810 IF P$(J,J)<>"*" THEN PRINT P$(J,J);
9820 NEXT J
9830 POSITION 0,23:PRINT :PRINT :PRINT :PRINT
9840 RETURN

9900 P$="Please talk sensibly !"
9910 RETURN
```



aktiv und kreativ computern

Die hier vorgestellten 21 Spiele sind speziell für den Atari geschrieben und auf den Modellen 600 XL und 800 XL getestet worden. Spannung, Action und bewegte Grafik stehen im Vordergrund. Das Buch zeigt einmal die enormen Möglichkeiten des Atari-BASIC, zum anderen versetzt es den Leser in die Lage, die Programme zu verstehen, zu analysieren und in eigene Programme einzubauen. Jedes Programm ist vollständig aufgelistet und erläutert; die Spielregeln und mögliche Erweiterungen werden ausführlich beschrieben.
Anhänger bewegter Grafik – Anfänger wie Fortgeschrittene – kommen voll auf ihre Kosten. Je intensiver man sich mit diesem Buch beschäftigt, um so deutlicher offenbaren sich die hier angewendeten raffinierten Programmiertechniken, die die außergewöhnlichen Fähigkeiten des Atari voll ausnutzen.



ISBN 3-8023-0788-7